پروتکلی جدید براي broadcast در محیط MPLS و گسترش آن براي پشتیباني از multicast در محیطهای MPLS *

سیاوش صمدیان برزکی s.samadian@ece.ut.ac.ir
مظفر بگ محمدی
ناصر یزدانی nasyaz@sofe.ece.ut.ac.ir

گروه برق و کامپیوتر دانشکده فنی
دانشگاه تهران، آزمایشگاه روتر

چکیده

با توجه به افزایش کاربردهاي multicast و تمایل به ارائه سرویسهاي مربوط به آن ، لازم است هر تکنولوژي جدیدي از آن پشتیباني کند. در حال حاضر تکنولوژي MPLS براي ارسال سریع بسته هاي IP و ایجاد سرویسهاي جدید در شبکه جهاني در حال شکل گیري و استاندارد شدن است. متأسفانه ، پشتیباني از multicast در این محیط داراي مشکلات و موانع متعددي است. مشکل اصلي براي پشتیباني از پروتکلهاي مسیریابي حالت مجتمع multicast مانند DVMRP و PIM-DM ، نبودن یک مکانیزم broadcast در محیط MPLS مي باشد. ما در این مقاله براي اولین بار به ارائه یک پروتکل جدید براي broadcast بصورت بهینه در محیط MPLS مي پردازیم. روش ارائه شده از یک نود مرکزي بنام BLAC (Broadcast Label Assignment Center) براي تخصیص برچسب استفاده مي کند. با گسترش این پروتکل ما پروتکل جدیدي را براي پشتیباني از گروههاي مجتمع multicast در MPLS ارائه کرده ایم که از نظر میزان برچسب مصرفي و همچنین میزان حافظه اشغالی توسط جداول ارسال نیز نسبت به مکانیزمهاي دیگر ارائه شده کاراتر است.

۱ مقدمه

بدون تردید multicast (ارسال داده ها از یک نقطه به چند نقطه یا از چند نقطه به چند نقطه دیگر) به یک موضوع داغ پژوهشی تبدیل شده [۷] و در بسیاری از شبکه های کنونی بعنوان یک قابلیت مهم وجود دارد [۳][۶]. همچنین برای عرضه کنندگان محصولات شبکه ، پشتیبانی از آن بعنوان یک مشخصه بسیار مطلوب مطرح می باشد [۷]. multicast در واقع تکنیکی است که ما را قادر می سازد تا با استفاده از یک ارسال بازای هر بسته ، داده های مورد نظر را به صورت کارا به چندین مقصد که در واقع یک گروه multicast را تشکیل می دهند برسانیم. اگربخواهیم همین کار را با استفاده از unicast انجام بدهیم لازم است که منبع به هر یک از دریافت کننده ها ، بصورت جداگانه ازهر بسته یک کپی بفرستد. بدیهی است که فواید multicast در این مورد اتلاف کمتر پهنای باند و منابع شبکه ، ایجاد موازی سازی در شبکه ، کاهش بار فرستنده و کاهش ترافیک شبکه می باشد[۳][۲].

ازجمله کاربردهای multicast می توان به کنفرانسهای صوتی و تصویری (مانند نشستهایی که در IETF برگزار می شوند) [۸][۳۲] ، پخش برنامه های تلویزیونی و رادیویی ، آموزش از راه دور (مثل شرکت در یک کلاس درس) و برگزاری نشست های از راه دور مربوط به انجام یک پروژه (هنگامی که اعضای پروژه از یکدیگر دور هستند) [۲۱] اشاره کرد. علاوه بر این موارد می توان از رد و بدل کردن نقشه های آب و هوایی بین پایگاههای هواشناسی [۹] ، به روز کردن تمام کپی های یک پایگاه داده ها یا جستجو کردن در تمام کپی های آن [۲][۳][۶] ، پخش کردن نتایج میانی مربوط به یک محاسبات توزیع شده به تمام نودهای شرکت کننده در آن [۲] ، توزیع نسخه جدید یک نرم افزار ، سرویسهای اطلاعاتی بورس سهام و کشف کردن منابع [۳][۶] نام برد.

با توجه به افزایش روز افزون کاربردهای multicast و گرایش رو به گسترش ISP ها برای ارائه سرویسهای مربوط به آن ، لازم است که هر تکنولوژی جدیدی در زمینه شبکه پشتیبانی از multicast را حتماً بصورت مطلوب مد نظر داشته باشد. از جمله این تکنولوژیها MPLS است که در حال حاضر توسط IETF در حال استاندارد شدن می باشد [۱۹]. در MPLS هدف این است که پیچیدگی را از روترهای داخل یک محدوده به روترهای ورودی و خروجی روی مرز این محدوده منتقل کنند تا عمل ارسال بسته ها در داخل محدوده MPLS با سرعت بیشتری انجام گرفته و در ضمن ، تضمین QoS هم در سرعتهاي بالا عملي گردد . با وجود اینکه در مورد ارسال بسته های unicast در MPLS مشکل چندانی وجود ندارد ، روش برخورد MPLS با ترافیک multicast هنوز راه حل مشخصي ندارد و احتیاج به تحقیقات و بررسیهای بیشتری دارد و در معماري MPLS نیز پشتیباني از آن تعریف نشده و موکول به پیشرفت مطالعات شده است [۱۹]. حل مشکل پشتیبانی از multicast در MPLS مطمئناً می تواند نقش مؤثری دربکارگرفته شدن این تکنولوژی درشبکه های آینده داشته باشد.

در این مقاله دو پروتکل جدید ارائه خواهد شد. پروتکل اول براي اولین بار broadcast را در محیط MPLS بصورت بهینه ممکن مي سازد. روش ارائه شده بر اساس چک کردن RPF (Reverse Path Forwarding) که در شبکه هاي IP مورد استفاده قرار مي گیرد مي باشد [۲][۱۳][۲۷][۲۸]. در این پروتکل به هر منبع ارسال داده broadcast بر اساس تقاضاي آن از یک نود مرکزي بنام BLAC یک برچسب اختصاص داده مي شود. از آن ببعد ، این برچسب نمایانگر بسته هاي broadcast ارسالي توسط منبع مي باشد و هر LSR (Label Switch Router) با استفاده از آن و بر اساس RPF ، واسط ورودي صحیح را که از طریق آن منتظر بسته هاي broadcast منبع

* قسمتي از هزینه هاي مربوط به این کار توسط مرکز تحقیقات مخابرات ایران تأمین شده است.

مي باشد تعيين مي نمايد. اگر بسته هاي مربوطه از طريق اين واسط به LSR برسند ، آنها را از طريق تمام واسطهاي ديگر خود ارسال مي كند. در غير اينصورت اين بسته ها را دور ميانداز. براي اينكه فضاي برچسب broadcast در سرآيند (header) لايه MPLS مشخص شود ، پيشنهاد مي كنيم كه يک مقدار خاص در سرآيند لايه ۲ (مثلاً در فيلد *ethertype* براي Ethernet و IEEE 802.3 يا در فيلد *Protocol* براي PPP) تعريف شود. اگر اين راه حل امكان پذير نبود ، راه حل دوم ما اين است كه يک مقدار برچسب خاص تعريف شود كه در بالاي پشته برچسب (label stack) قرار گيرد و برچسب تخصيص داده شده به منبع در سطح بعدي پشته قرار داده شود. به اين شيوه فضاهاي برچسب unicast و broadcast از هم جدا مي شوند و براي انجام broadcast توسط هر منبع تنها يک برچسب از فضاي برچسب broadcast مصرف مي شود.

با استفاده از اين پروتكل ما يک پروتكل جديد نيز براي پشتيباني از گروههای مجتمع multicast در محيط MPLS ارائه داده ايم كه شبيه پروتكل DVMRP [۱۳][۱۸] مي باشد. اين پروتكل نسبت به پروتكلهاي ارائه شده موجود برچسب كمتري مصرف مي كند و داراي جداول ارسال كوچكتري است. دلايل برتری روشهای ما بصورت تحليلی در مقاله آمده است.

در بخش بعدی ابتدا مفاهيم لازم در زمينه multicast بطور مختصر توضيح داده خواهد شد. در بخش ۳ نيز مفاهيم مورد نياز و اوليه MPLS مورد بررسی قرار می گيرند. در بخش چهارم بعضي مشكلات پياده سازی multicast در MPLS را كه راه حلهايي براي آنها وجود دارد شرح خواهيم داد. در بخش ۵ ما به ارائه پروتكل broadcast خود مي پردازيم. سپس در بخش ۶ ، پروتكل ارائه شده براي پشتيباني از گروههاي مجتمع multicast ارائه شده است. مسائل جانبی موجود در پروتكلهاي ارائه شده در بخش ۷ آمده اند. در بخش ۸ كارهای ارائه شده ديگران را در زمينه پشتيباني از multicast در محيط MPLS بررسي مي كنيم و آنها را با پروتكلهاي ارائه شده خود مقايسه مي كنيم. بخش ۹ نيز به جمع بندی مطالب ارائه شده و قدمهای آينده ما اختصاص دارد.

۲ Multicast

در multicast داده برای گروهی از كاربران ارسال می شود. گروههاي multicast را از لحاظ توزيع مكاني كاربران مي توان به دو نوع عمده تقسيم نمود:

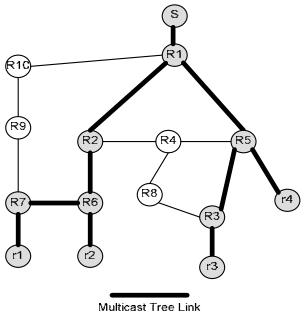
۱. **گروههاي مجتمع:** در اين حالت اعضاي گروه به صورت مجتمع توزيع شده اند، يعني اكثر نودهاي شبكه عضو گروه multicast مورد نظر مي باشند.
۲. **گروههاي پراكنده:** در اين حالت اعضاي گروه به صورت پراكنده توزيع شده اند، يعني اكثر نودهاي شبكه داراي عضو براي گروه multicast مورد نظر نمي باشند.

برای توزيع داده multicast مربوط به يک گروه، يک درخت multicast تشكيل شده و بسته به روی تمام شاخه های اين درخت ارسال می گردد. يک درخت multicast بايد در برگيرنده تمام اعضاي گروه باشد. اما فرستنده ها مي توانند عضو گروه نباشند و فقط داده را براي اعضاي گروه مورد نظر بفرستند. در حالت مجتمع فرض بر اين است كه تمام نودهاي شبكه عضو گروه مورد نظر هستند، مگر اينكه عدم علاقمندي خويش را اعلام كنند. براي گروههاي پراكنده، وضع برعكس مي باشد [۱۳][۷][۲][۱۱].

مقالات متعددی وجود دارد كه به مقايسه درختهای مختلف توزيع از جنبه های مختلف و مزايا و معايب پروتكلهای ارائه شده برای گروههای مجتمع و پراكنده می پردازند كه از بين آنها می توان به [۹] ، [۱۰] و [۱۱] اشاره كرد.

۲,۱ گروههاي مجتمع

پروتكل DVMRP (Distance Vector Multicast Routing Protocol) [۱۸] اولين پروتكل ارائه شده برای multicast می باشد كه براي پشتيباني از گروههاي مجتمع طراحي شده است. در اين پروتكل به ازاي هر فرستنده S و هر گروه G يک درخت منبع (S, G) ساخته مي شود. منظور ما از درخت منبع درختي است كه ريشه آن فرستنده مي باشد و شامل كوتاهترين مسيرها بين منبع و هر گيرنده مي باشد. در شكل شماره ۱ مثالي از درخت منبع نشان داده شده است. روش ساخت اين درخت بدين صورت است كه وقتي S اولين بسته خود را به سمت گروه G مي فرستد، روتر متصل به S، بسته را روی تمام واسطهای خروجی خود ارسال می كند. سپس هر روتری كه اين بسته را می گيرد چک می كند تا بسته هايی را كه روی ورودی مناسب رسيده اند ، تشخيص دهد. ورودی مناسب به آن گفته می شود كه دارای كوتاهترين فاصله بين روتر و منبع S باشد (يعنی كوتاهترين مسير در جهت معكوس ارسال بسته). بسته هايی كه روی ورودی های مناسب رسيده اند به تمام خروجی های روتر (بجز پورتی كه بسته روی آن آمده است) فرستاده می شوند. اين عمل را RPF (Reverse Path Forwarding) می گويند [۲][۲۷][۲۸] و در پايين آوردن بار شبكه بسيار حائز اهميت است زيرا روترها يک بسته را ممكن است از چند ورودي دريافت كنند و مسلماً ارسال همه آنها موجب اتلاف پهناي باند شبكه خواهد شد. بدين صورت بسته ها در تمام شبكه پخش می شوند. در مرحله بعد ، چنانچه در شبكه متصل به يک روتر ، هيچ host عضو گروه multicast نباشد ، روتر مربوطه يک پيغام حذف بسمت منبع می فرستد و روتر بعدي كه در طول مسير اين پيغام را می گيرد حالت واسط مربوطه را در جدول مسيريابی multicast به حالت حذف تغيير می دهد و يک زمان سنج (timer) به آن اختصاص می دهد. تا پايان مهلت اين زمان سنج، بسته های منبع S به اين واسط فرستاده نخواهند شد [۷][۱۳][۱۸]. همچنين اگر اين روتر از تمام واسطهاي خروجي خود چنين پيغامي را دريافت كند، يعني تمام واسطهاي خروجي آن در حالت حذف باشند، پيغام حذف را به سمت منبع خواهد فرستاد. علاوه بر آن ، يک شاخه حذف شده از درخت را با ارسال يک پيغام Graft مي توان دوباره برقرار كرد. به اين مكانيسم پخش وحذف انتخابي (flood & prune) مي گويند، يعني ابتدا داده multicast در تمام شبكه پخش مي شود سپس نودهايي كه به داده علاقه ندارند پيغام حذف مي فرستند.



شكل ۱: درخت منبع براي S

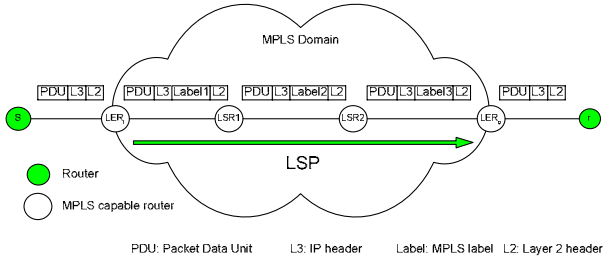
مشكل اصلي پياده سازي DVMRP در MPLS، عدم وجود مكانيسم broadcast در MPLS مي باشد. اين مكانيسم در مرحله پخش مورد نياز مي باشد و ما در اين مقاله با معرفي يک الگوريتم جهت broadcast، مشكل فوق را مرتفع كرده ايم.

۳ MPLS

عمل ارسال بسته ها در حالت عادی در لایه IP انجام می شود و احتیاج به پردازش یک سرآیند نسبتاً بزرگ و انجام یک الگوریتم تطبیق بزرگترین پیشوند (prefix matching) دارد که عملی زمانگیر است. اما در MPLS ارسال بسته ها با استفاده از روش تعویض برچسب و براساس تطبیق کامل یک برچسب کوچک بسادگی و با زمان کمتر انجام پذیراست[۲۰]. [۲۲] میزان کارآیی MPLS را در شبکه های backbone ارزیابی می کند. نتایج ارزیابی نشان می دهد که برای تجهیزات MPLS که خوب طراحی شده باشند ، تأخیر اصلی بسته های IP می تواند به اندازه ۳۰% یا بیشتر کاهش یابد.

یک ناحیه MPLS ، از روترهایی که قابلیت پشتیبانی از MPLS را دارند تشکیل می شود. به روترهایی که در هسته ناحیه قرار دارند LSR (Label Switching Router) ، و به روترهایی که در مرز ناحیه قرار دارند LER (Label Edge Router) می گویند [۱][۱۹][۲۰]. بدین صورت که ابتدا در LER های ورودی ناحیه MPLS با استفاده از پروتکلهای عادی مسیریابی مانند OSPF و BGP تعیین می شود که بسته ها از چه مسیری باید بروند. این اطلاعات مسیریابی ، فضای ارسال بسته ها را به کلاسهای هم ارزی هدایت یا FEC (Forwarding Equivalence Class) تقسیم می کنند. بدین صورت که یک مجموعه از بسته ها که باید از یک مسیر یکسان بروند ، متعلق به یک FEC خواهند بود و لذا بطور یکسان ارسال می شوند [۲۰][۱][۱۹].

به هر FEC یک برچسب کوتاه و با طول ثابت ۲۰ بیت اختصاص داده می شود. این برچسب می تواند در فضاهای موجود مورد استفاده درسرآیند لایه ۲ قرار داده شود (مثل استفاده ازفضای VPI/VCI در ATM) یا اینکه بصورت یک سرآیند مخصوص بین سرآیند های لایه ۲ و ۳ قرار گیرد. پس از ارسال بسته های برچسب خورده از LER به LSR بعدی، این برچسب بعنوان اندیس یک جدول ارسال (در LSR) استفاده می شود و از روی آن برچسب خروجی جدید و پورت خروجی مناسب برای رسیدن به LSR بعدی بدست می آید. سپس برچسب قبلی با برچسب جدید عوض می شود و بسته به LSR بعدی ارسال می گردد. این عمل تا رسیدن بسته به LER خروجی ادامه پیدا می کند و در آنجا با برداشتن برچسب ، بسته به حالت قبل از ورود به ناحیه MPLS تبدیل می شود و به دنیای خارج از MPLS تحویل می گردد [۲۰][۱][۱۹]. این روش باعث می شود که عمل lookup درلایه ۳ تنها به LERها محدود شود و در LSRهای داخل ناحیه MPLS عمل ساده تر تعویض برچسب انجام شود [۲۰]. به دنباله ای از برچسب ها که از LER ورودی شروع و به LER خروجی ختم می شود و در ناحیه MPLS تشکیل یک مسیر را می دهند، یک LSP (Label Switched Path) می گویند [۱۹][۱]. در شکل شماره ۲ مفاهیم MPLS نشان داده شده است.

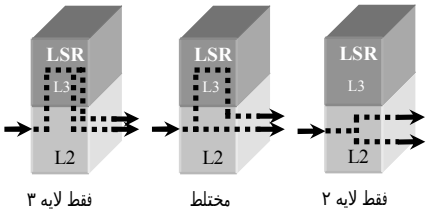


شكل ۲ : مفاهيم MPLS

۴ مشكلات پياده سازی multicast در MPLS

۴,۱ انواع ارسال

از آنجاكه براي ترافيک unicast تنها يک واسط ورودي و يک واسط خروجي در LSR وجود دارد ، لذا اين ترافيک يا در لايه ۲ يا در لايه ۳ مي تواند سوييچ شود. اما بدليل اينكه تعداد واسطهاي خروجي ترافيک multicast بيش از يكي مي باشد ، مي توان حالتي را در نظر گرفت كه ترافيک براي بعضي از شاخه ها در لايه ۲ و براي بعضي ديگر در لايه ۳ سوييچ شود. بنابراين عمل ارسال بسته های multicast در روترهای ناحيه MPLS می تواند به سه صورت انجام شود [۱۷] كه در شكل ۳ نشان داده شده اند. روشهاي ارائه شده ما در اين مقاله احتياج به پشتيباني از ارسال مختلط دارند



شكل ۳: انواع ارسال

۴,۲ مشخصات درختهای IP multicast

بعضي از مشخصات درختهای IP multicast در يک ناحيه MPLS مسائلي ايجاد مي نمايند كه در اين قسمت به توضيح آنها مي پردازيم.

۱. **تجمع (aggregation):** در حالت unicast می توان یک برچسب را به چندین آدرس مقصد متفاوت نسبت داد. یعني یک ورودی در جدول مسیریابی (یا یک FEC) و لذا یک LSP برای آن داشت [۲۰][۱۹][۱]. این تکنیک موجب صرفه جويي در مصرف برچسبها مي گردد. اما تحقق این امر برای درختهای multicast با آدرسهای مقصد multicast متفاوت در دست مطالعه می باشد [۱۶][۱۷]. روشهايي در این مورد ارائه شده است که در بخش ۸ به آنها اشاره خواهد شد.

۲. **پخش و حذف انتخابی:** همانطور که در بخش ۲ دیدیم ، در بعضی از پروتکلها عمل پخش و حذف انتخابی بصورت متناوب تکرار می شود [۱۳]. در این حالت ساختار درختی بدست آمده در لایه ۳ بسیار متغیر می باشد و نگاشت آن به یک p2mp LSP (point to multipoint) برچسب زیادی مصرف خواهدکرد. همچنین چون این درختها با پخش ترافیک در شبکه ایجاد می شوند، لذا برای برپایی LSP احتیاج به یک روش سریع و با سیگنالهای بالاسري (overhead) کم دارند [۱۶][۱۷]. هر دو مشکل ذکر شده در پروتکل ارائه شده در این مقاله بصورت مؤثر حل شده اند. تشکیل درخت در این پروتکل توسط مکانیزم broadcast پشتيباني مي شود. علاوه بر آن بازاي هر درخت multicast تنها یک برچسب از محدوده فضاي برچسب multicast مصرف مي شود و فضاي برچسب unicast دست نخورده باقي مي ماند. این باعث مي شود که مصرف برچسب توسط این پروتکل مینیمم باشد.

از آنجا که برای پروتکلهاي حالت مجتمع ، در هر LSR ایجاد و حذف یک مسیر لایه ۳ توسط ترافیک trigger مي شود ، به نظر مي رسد که برقراري و از بین بردن LSP متناظر با آن مسیر نیز بهتر است بر اساس ترافیک صورت بگیرد. علاوه بر آن اگر قابلیت ارسال مختلط وجود نداشته باشد ، trigger شدن بر اساس ترافیک نیاز به این خواهد داشت که LSR بالادستي (upstream) ایجاد یک LSP را آغاز کند (روش Downstream on Demand یا Upstream Unsolicited) [۱۶][۱۷]. یک روش توزیع برچسب بالادستي (upstream label distribution) براي پشتيباني از گروههاي multicast حالت مجتمع وجود دارد که در بخش ۸ به آن اشاره خواهد شد. روش ما از هیچ شیوه LSP trigger استفاده نمي کند و لذا مسائل ذکر شده در مورد آن مطرح نیست.

برای اطلاع از سایر مشکلات موجود برای پشتیبانی از multicast در محیط MPLS به [۱۶] و [۱۷] مراجعه شود.

۵ روش بهینه broadcast در MPLS

براي اینکه داده بتواند با استفاده از روش سوییچ شدن برچسب broadcast شود ، لازم است تا داده هاي broadcast به نحوي مشخص شوند. این کار مي تواند با تعریف یک مقدار خاص در سرآیند لایه ۲ (مثلاً در فیلد *ethertype* براي Ethernet و IEEE 802.3 یا در فیلد *Protocol* براي PPP) انجام گیرد تا از روي آن فضاي برچسب broadcast در سرآیند MPLS تشخیص داده شود. اگر این راه حل امکان پذیر نبود ، راه حل دوم ما این است که یک مقدار برچسب خاص از بین برچسبهاي MPLS تعریف شود که در بالاي پشته برچسب (label stack) قرار گیرد. با دیدن این برچسب خاص ، هر LSR مي فهمد که برچسب موجود در سطح بعدي پشته مربوط به فضاي برچسب broadcast مي باشد. راه حل دوم بخاطر این ۴ بایت برچسب اضافي باعث مصرف بیشتر پهناي باند شبکه مي شود. با وجود اینکه ما راه حل اول را توصیه مي کنیم ، براي دستیابي به یک شیوه مستقل از لایه ۲ ما در ادامه مقاله بر اساس راه حل دوم عمل مي کنیم. لذا هنگامیکه راه حل اول براي یک تکنولوژي لایه ۲ ممکن بود ، لازم است تا تبدیلات لازم بطور مناسب در الگوریتمهاي ارائه شده صورت بگیرند.

بدینوسیله هر LSR با دیدن برچسب خاص ذکر شده در بالاي پشته ، داده را به شیوه broadcast ارسال مي کند. ساده ترین راه براي ارسال داده با برچسب خاص broadcast براي یک LSR این است که آنرا به تمام واسطهاي دیگر خود بجز واسطي که بسته را از آن گرفته ارسال نماید. این کار با اینکه عمل broadcast را انجام می دهد اما بسیار غیر بهینه است و ترافیک بسیار سنگینی را متحمل شبکه می کند که مي تواند باعث ایجاد congestion در شبکه شود. علت این امر آن است که بسته های تکراری روی لینکهای بین هر دو نود عبور خواهند کرد و علاوه بر آن ممکن است به هر نود نهایی بیش از یک نسخه از داده برسد که لازم است در لایه های بالاتر آنرا فیلتر کند. این باعث می شود که هم پهنای باند زیادی مصرف شود و هم بار پردازشی نودها زیاد باشد [۲۷][۲۸]. اگر فرض کنیم که برچسب خاص broadcast برابر X باشد ، عملیات انجام شده در هر LSR را با رسیدن بسته اي با برچسب X بصورت زیر مي توان نمایش داد :

IF $Label_{in}=X$ ***THEN***
  $TTL \leftarrow TTL - 1;$
  IF $TTL = 0$ ***THEN*** $OutgoingLinkSet \leftarrow \{\};$
  ELSE $OutgoingLinkSet \leftarrow AllLinks - IncomingLink;$
  ENDIF
ELSE $OutgoingLinkSet \leftarrow \{ILM[Label_{in}].Interface_{out}\};$
  Change $Label_{in}$ *with* $ILM[Label_{in}].Label_{out};$
  Forward *the revised packet to OutgoingLinkSet;*
ENDIF

همانطور که مي بینیم ، این الگوریتم براي اینکه داده هاي broadcast تا ابد در شبکه نمانند ، احتیاج به کم کردن فیلد TTL دارد.

روشهای بهتری هم برای عمل broadcast در شبکه های IP وجود دارد که در [۲۷] ، [۲] و [۲۸] بررسی شده اند. از جمله این روشها الگوریتمهای RPF ، Extended RPF و RPB (Reverse Path Broadcasting) می باشند که همگی از آدرس مبدأ بسته برای تعیین واسطهای خروجی استفاده می کنند. اما می دانیم که در سرآیند بسته MPLS تنها یک برچسب وجود دارد که هیچ ارتباطي با مبدأ یک بسته ندارد و یک LSR تنها با مراجعه به سرآیند لایه IP می تواند مبدأ بسته را کشف کند. لذا براي اینکه بتوانیم بدون مراجعه به سرآیند لایه ۳ این روشها را در MPLS نیز استفاده کنیم ، باید بتوان به شیوه اي از روي برچسب موجود در سرآیند MPLS به هویت

فرستنده بسته پي برد. بنابراين به اين شيوه عمل مي كنيم كه هر منبع S كه بخواهد داده broadcast ارسال كند ابتدا از يک LSR مركزي بنام BLAC (Broadcast Label Assignment Center) تقاضاي يک برچسب broadcast مي نمايد. اين نود ، مسئول كنترل تخصيص و يا رها كردن يک برچسب از محدوده برچسبهاي broadcast به هر منبع متقاضي ارسال داده broadcast مي باشد. براي اين كار ، S يک پيغام *Broadcast_Label_Request* را به آدرس BLAC ارسال مي نمايد. نود BLAC با ديدن اين پيغام به صورت زير عمل مي كند :

IF *(∃ $Label_B \in Label\_Range_B$)* ***THEN***
  Send *a Broadcast_Label_Assignment message(X, $Label_B$) destined to S to all interfaces;*
  $Label\_Range_B \leftarrow Label\_Range_B - \{Label_B\}$;
  Insert *(IP_S, $Label_B$) in $Table_{Assigned\_Labels}$;*
ELSE
  Discard *the Broadcast_Label_Request message;*
ENDIF

اين الگوريتم به اين شيوه عمل مي كند كه ابتدا نود BLAC نگاه مي كند تا ببيند آيا برچسبي در مجموعه برچسبهاي broadcast ($Label\_Range_B$) مانده يا نه. در صورت وجود برچسب تخصيص داده نشده $Label_B$ ، پيغام زير به تمام واسطهاي نود BLAC ارسال مي شود تا در شبكه broadcast شود :

	Shim Header	Source Address	Destination Address	Payload
L2 header	X	$Label_B$	IP_{BLAC}	IP_S	Broadcast_Label_Assignment

بدين وسيله نود BLAC به تمام LSR هاي موجود در حوزه MPLS از جمله خود منبع S اطلاع مي دهد كه از آن به بعد ، برچسب $Label_B$ به S تعلق دارد و S مي تواند از اين برچسب ، براي ارسال داده هاي broadcast خود استفاده كند. برچسب X كه در بالاي پشته برچسبها قرار داده مي شود يک برچسب خاص است كه تمام LSR ها مي شناسند و براي ارسال داده هاي broadcast مربوط به نود BLAC استفاده مي شود. هنگاميكه اين پيغام به منبع S مي رسد ، منبع براي ارسال داده هاي خود ابتدا $Label_B$ را در پشته برچسب هر بسته قرار مي دهد و سپس برچسب خاص Y را كه همه LSR ها مي شناسند در بالاي پشته قرار مي دهد. بسته داده broadcast بدست آمده ، توسط منبع S از طريق تمام واسطها ارسال مي شود. بسته هاي داده broadcast منبع S بدين صورت خواهند بود :

	Shim Header	Source Address	Destination Address	Payload
L2 header	Y	$Label_B$	IP_S	$IP_{Broadcast}$	Broadcast Data

در هر LSR با ديدن بسته اي با برچسب Y در بالاي پشته ، مي دانيم كه برچسب بعدي موجود در پشته مربوط به عمل broadcast مي باشد. توجه كنيد كه برچسب Y هنگاميكه در سرآيند لايه ۲ هيچ مقدار مشخص كننده اي براي تشخيص نوع بسته وجود ندارد لازم است ، زيرا در صورت عدم وجود آن ، بايد محدوده برچسبها را به دو قسمت محدوده برچسبهاي broadcast و محدوده برچسبهاي Unicast تقسيم كنيم كه اين امر باعث كاهش فضاي برچسبهاي Unicast مي شود و خوب نيست. ما به اين شيوه دو فضاي برچسب Unicast و broadcast را از هم مستقل كرديم و لذا هر دو مي توانند از تمام فضاي برچسب استفاده كنند. در عوض مقدار پهناي باند مصرفي داده هاي broadcast بيشتر خواهد شد و علاوه بر آن پردازش لازم در هر LSR نيز بازاي هر بسته داده broadcast بيشتر مي شود. ما معتقديم هزينه هاي ياد شده در برابر سود ناشي از مستقل بودن دو فضاي برچسب ناچيز است.

روش برخورد LSR ها با بسته هاي با برچسب Y بدين صورت است :

IF *top label is Y* ***THEN***
  Pop *the top label;*
  IF *$Label_B$ exists in BFT* ***THEN***
    $iif_{RPF} \leftarrow BFT(Label_B).interface;$
    IF *$iif \neq iif_{RPF}$* ***THEN***
      Discard *the packet;*
    ELSE
      Push *label Y at top of stack in the packet;*
      Forward *the packet to AllLinks – {iif};*
    ENDIF
  ELSE
      Push *label Y at top of stack in the packet;*
      Forward *the packet to AllLinks – {iif};*
  ENDIF
ENDIF

در اين الگوريتم ابتدا چک مي شود كه آيا برچسب دوم (بعد از برچسب Y) در جدولي بنام BFT (Broadcast Forwarding Table) موجود هست يا نه. اين جدول كه براي ارسال داده هاي broadcast در هر LSR نگهداري مي شود با استفاده از پيغامهاي broadcast شده نود BLAC با برچسب X ساخته مي شود. هر entry اين جدول شامل برچسب broadcast اختصاص داده شده به هر منبع و واسطي از LSR است كه با توجه به جدول lookup ، كوتاهترين مسير تا منبع از طريق آن مي باشد. اين واسط در الگوريتم ، برابر iif_{RPF} قرار داده شده است. چگونگي ساخت اين جدول را در ادامه هنگام پردازش بسته هاي برچسب خورده با برچسب X خواهيم ديد. اگر برچسب $Label_B$ در جدول BFT وجود داشت (يعني LSR ، قبل يا همزمان با اينكه منبع S از برچسب اختصاصي خود مطلع شود ، از تخصيص آن به S مطلع شده و آنرا در جدول خود قرار داده است) ، از الگوريتم RPF كه قبلاً ذكر شد ، براي ارسال داده broadcast استفاده مي شود. همانطور كه در الگوريتم نيز ديده مي شود ، در روش RPF اگر داده روي واسطي آمده باشد كه كوتاهترين مسير از LSR تا نود فرستنده داده (در جهت معكوس داده) از طريق آن باشد ، داده را روي تمام واسطهاي ديگر بجز واسطي كه از آن آمده است (*AllLinks – iif*) ارسال مي كنيم و در غير اينصورت آنرا دور مي اندازيم [۲۷][۲][۲۸]. ممكن است $Label_B$ در جدول BFT وجود نداشته باشد. اين حالت وقتی پيش می آيد كه پيغام *Broadcast_Label_Assignment* شامل تخصيص برچسب $Label_B$ به S هنوز به LSR نرسيده اما به منبع S رسيده و داده هاي ارسالي از S با آن برچسب نيز به LSR رسيده اند. در اين حالت چند شيوه را مي توان در نظر گرفت. يا اينكه داده آمده را دوربيندازيم كه اين باعث كاهش قابليت اطمينان عمل broadcast مي شود. پيشنهاد ما اين است كه بدون چک كردن RPF داده را به تمام واسطها بجز واسطي كه از آن آمده است ارسال

کنیم. در این حالت LSR تا موقعي که اطلاعات لازم براي ارسال بهینه داده را در اختیار ندارد از روش غیر بهینه broadcast استفاده مي کند که باعث اتلاف پهناي باند شبکه مي شود اما قابلیت اطمینان را بالا مي برد. قابل توجه است که مدت زمان استفاده از الگوریتم غیر بهینه کوتاه و محدود به رسیدن اطلاعات از نود BLAC مي باشد و هزینه پرداخت شده مي تواند ناچیز باشد. روش دیگر این است که بسته را با استفاده از سرآیند لایه IP و بصورت بهینه با چک کردن RPF اما در لایه ۳ ارسال کنیم. این امر باعث افزایش پردازش LSR مي شود.

هنگامیکه منبع S دیگر احتیاج به برچسب broadcast ندارد ، این امر را به نود BLAC اعلام می کند و برچسب مورد نظر آزاد می شود. براي این کار ، S یک پیغام *Broadcast_Label_Release* را بسمت نود BLAC ارسال مي کند و نود BLAC برچسب مربوطه را آزاد مي کند. اما براي اینکه تمام LSR هاي موجود در حوزه MPLS نیز بدانند که دیگر برچسب مربوط به منبع S اعتبار ندارد و آنرا از جداول خود حذف کنند ، نود BLAC پیغام زیر را در تمام حوزه broadcast مي کند :

	Shim Header	Source Address	Destination Address	Payload
L2 header	X	$Label_B$	IP_{BLAC}	IP_S	Broadcast_Label_Release

توجه داشته باشید که در این حالت خود منبع S هم از موفقیت عمل آزاد شدن برچسب مربوطه آگاه مي شود. ممکن است پیغامهاي *Broadcast_Label_Request* و *Broadcast_Label_Release* در شبکه بدلیلي گم شوند و به نود BLAC نرسند ، یا اینکه پیغامهاي *Broadcast_Label_Request* بدلیل عدم وجود برچسب broadcast در محدوده برچسبهاي broadcast توسط نود BLAC دورانداخته شوند. لذا منبع S با ارسال هر یک از دو پیغام مذکور ، یک زمان سنج را نیز برابر α *RTT_{BLAC} قرار مي دهد. با پایان پذیرفتن مقدار زمان سنج ، اگر پیغامهاي معادل *Broadcast_Label_Assignment* و *Broadcast_Label_Release* هنوز از نود BLAC به منبع S نرسیده باشد ، پیغامهاي اولیه دوباره توسط منبع S بسمت نود BLAC ارسال مي شوند. این عمل بهمین صورت براي پیغامهایي که بعداً دوباره ارسال می شوند اما با استفاده از شیوه exponential back-off (که در TCP و Ethernet نیز موجود است [۳۳]) ادامه پیدا مي کند. مقدار RTT_{BLAC} زمان رفت و برگشت (Round Trip Time) بین منبع و نود BLAC مي باشد. مقادیر ضریب α و RTT_{BLAC} نیز پارامترهاي طراحي هستند که باید از روي شبیه سازي مقدار بهینه آنها تعیین شود. بنابراین روش برخورد نود BLAC با پیغام *Broadcast_Label_Release* دریافت شده از S به این صورت است :

IF *($\exists$ $(IP_S , Label_B)$ in $Table_{Assigned\_Labels}$)* ***THEN***
  Send *a Broadcast_Label_Release message(X, $Label_B$) destined to S to all interfaces;*
  $Label\_Range_B \leftarrow Label\_Range_B + \{Label_B\}$;
  Delete *$(IP_S , Label_B)$ from $Table_{Assigned\_Labels}$;*
ELSE
  Discard *the Broadcast_Label_Release message;*
ENDIF

این الگوریتم دقیقاً برعکس الگوریتم ارائه شده براي پردازش پیغام *Broadcast_Label_Request* توسط نود BLAC مي باشد که قبلاً بررسي شد.

تا بحال فرض کردیم که تمام نودهاي شبکه از آدرس نود BLAC اطلاع دارند. حال مي خواهیم بدانیم که چگونه نود BLAC تعیین مي شود. ۳ راه حل وجود دارد. روش اول این است که مدیر شبکه (administrator) بصورت استاتیک و در ابتداي کار یک LSR خاص را بعنوان نود BLAC انتخاب کند و آنرا بعنوان نود BLAC تنظیم (configure) نماید و آدرس آنرا نیز روي تمام LSR ها وارد کند. این روش بسیار بد است چون میزان کار مدیر شبکه براي تغییر آن زیاد خواهد بود و احتمال اشتباه نیز بالا است. روش دیگري که مي توان بکار برد این است که یک LSR خاص توسط مدیر شبکه انتخاب شود و بصورت استاتیک بعنوان نود BLAC تنظیم شود. براي اینکه بقیه نودها نیز از آدرس آن با اطلاع شوند ، نود BLAC وظیفه دارد در فواصل زماني مشخص $T_{Broadcast\_Address}$ پیغام *BLAC_Address* را بصورت زیر به تمام واسطهاي خود ارسال کند تا در شبکه broadcast شود :

	Shim Header	Source Address	Destination Address	Payload
L2 header	X	S=1	IP_{BLAC}	$IP_{Broadcast}$	BLAC_Address

هنگامیکه یک LSR بسته اي با برچسب مشخص X دریافت مي کند که تنها برچسب موجود در بسته است (مقدار فیلد S=1 در برچسبهاي MPLS براي نشان دادن ته پشته بکار مي رود [۲۹]) ، مي فهمد که این بسته شامل آدرس نود BLAC است و لذا در صورتیکه نوع پیغام را از payload بسته *BLAC_Address* تشخیص داد ، از روي آدرس مبدأ در سرآیند IP آدرس نود BLAC را استخراج مي کند و entry زیر را به جدول BFT خود اضافه مي کند :

Label-in	Input interface	IP Address
X	Routing[IP_{BLAC}]	IP_{BLAC}

Routing[IP_{BLAC}] یک واسط خروجي را بر مي گرداند که از عمل lookup لایه ۳ براي IP_{BLAC} بدست مي آید. در این جدول همانطور که مي بینیم output interface مشخص نشده است و این به آن علت است که براي ارسال داده broadcast مجموعه واسطهاي خروجي همواره *AllLinks* – *Input interface* مي باشد. همچنین برچسب خروجي چون همواره برابر برچسب ورودي است ، در جدول نیامده است.

روش سوم براي تعیین نود BLAC که مانند انتخاب monitor در ring ها یا انتخاب ریشه براي تشکیل درخت پوشا در Ethernet مي باشد [۳۳] این است که تمام LSR ها هنگام bootstrap یک پیغام *BLAC_Address* را مانند آنچه در بالا دیدیم ارسال کنند. هر LSR با دیدن آن پیغام و با چک کردن RPF بررسي کند که آیا آدرس مبدأ بسته از آدرس IP_{BLAC} ذخیره شده که تا به حال مي داند کوچکتر است یا نه. این آدرس در ابتداي امر برابر آدرس خود LSR قرار داده مي شود. اگر کوچکتر بود این آدرس را جایگزین مقدار قبلي IP_{BLAC} کند و پیغام را به تمام واسطهاي خود بجز

واسطي كه پيغام را از آن گرفته ارسال كند. اگر آدرس IP_{BLAC} قبلي كوچكتر بود بايد پيغام دريافتي را دوربيندازد. بدين ترتيب پس از رد و بدل كردن پيغامهاي ارسالي اوليه و همگرا شدن آنها ، تمام LSR ها آدرس LSR با كوچكترين آدرس IP موجود در شبكه را به عنوان آدرس نود BLAC انتخاب خواهند كرد. از آن به بعد بايد نود BLAC تعيين شده در فواصل زماني $T_{Broadcast\_Address}$ پيغام *BLAC_Address* را دوباره broadcast نمايد تا entry مربوطه در جداول BFT را refresh كند. اين entry مانند آنچه در بالا نشان داده شد مي باشد. بازاي اين entry يک زمان سنج هم معادل $\beta * T_{Broadcast\_Address}$ قرار داده مي شود و با پايان پذيرفتن آن اگر پيغام *BLAC_Address* دريافت نشد دوباره خود LSR مانند ابتداي كار كانديداي انتخاب به عنوان نود BLAC مي شود. مقدار β و $T_{Broadcast\_Address}$ پارامترهايي هستند كه از روي شبيه سازي بايد بدست بيايند و متأثر از قابليت اطمينان عمل broadcast و ميزان تحمل پذيري خطا مي باشند.

در اين روش آخر چند نكته قابل بررسي است. اولاً اينكه هر LSR مي تواند بر اساس ملاكهاي خاصي مثل قدرت پردازشي خود تصميم بگيرد كه در عمل ابتدايي تعيين نود BLAC شركت كند يا نه. اين بدان معناست كه نود مذكور در اين فاز آدرس خود را به عنوان نود BLAC در پيغام *BLAC_Address* اعلام نمي كند و لذا IP_{BLAC} در ابتداي امر بايد بجاي آدرس خود LSR ، برابر يک آدرس default قرار داده شود كه قبلاً از طريق مدير شبكه تنظيم شده است. يا اينكه مدير شبكه مي تواند يک LSR خاص را بعنوان نود default تنظيم كند تا اين نود هميشه در عمل ابتدايي تعيين نود BLAC شركت كند. نكته بعدي اين است كه هر LSR كه در ابتدا در تعيين نود BLAC شركت مي كند مي تواند يک معيار خاص مشترک بين تمام LSR ها را در پيغام *BLAC_Address* خود قرار دهد تا LSR هاي ديگر از روي آن (بجاي ملاک كوچكتر بودن آدرس IP) تصميم بگيرند كه آدرس آنرا بايد جايگزين IP_{BLAC} قبلي بكنند يا نه.

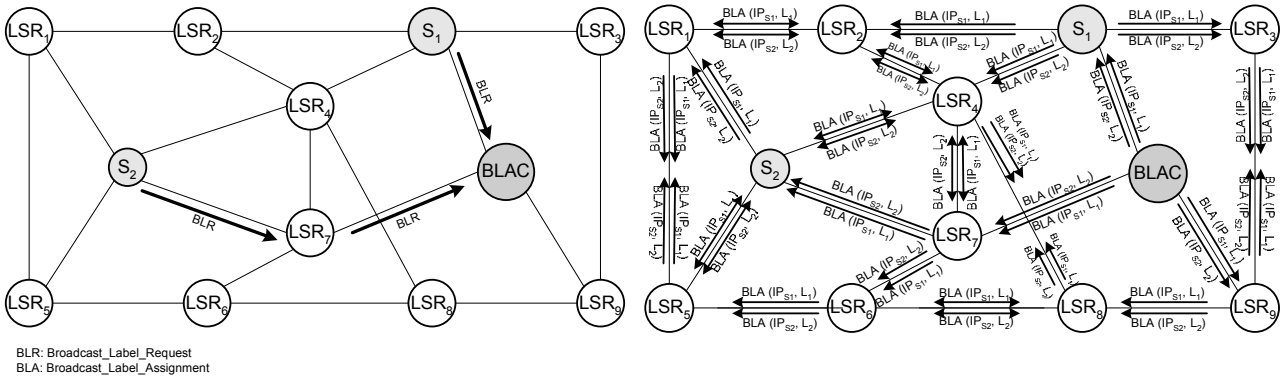
با توجه به مطالب ذكر شده تا بحال ، يک LSR در برخورد با بسته اي با برچسب X موجود در بالاي پشته به صورت زير رفتار مي كند :

```
IF top label is X THEN
   IF BFT(X) does not exist and S=1 THEN
      iif_RPF ← Routing[IP_BLAC];
   ELSE IF BFT(X) does not exist and S=0 THEN
      Discard the packet;
      Return;
   ELSE
      iif_RPF ← BFT(X).input_interface;
   ENDIF

   IF iif ≠ iif_RPF THEN
      Discard the packet;
   ELSE IF S=0 THEN
      Pop the top label;

      IF Payload is Broadcast_Label_Assignment THEN
         Insert (Label_B , Routing[IP_S] , IP_S) in BFT;
      ELSE IF Payload is Broadcast_Label_Release THEN
         Delete (Label_B , Routing[IP_S] , IP_S) from BFT;
      ELSE
         Discard the packet;
         Return;
      ENDIF

      Push label X at top of stack in the packet;
      Forward the packet to AllLinks – {iif};
   ELSE IF Payload is BLAC_Address THEN
      Insert (X , iif_RPF , IP_BLAC) in BFT;
      Forward the packet to AllLinks – {iif};
   ELSE
      Discard the packet;
   ENDIF
ENDIF
```

هر LSR در برخورد با هر پيغام *Broadcast_Label_Assignment* كه شرايط موجود در كد بالا را برآورده كند entry زير را در BFT اضافه مي كند :

Label-in	Input interface	IP Address
$Label_B$	$Routing[IP_S]$	IP_S

در شكل شماره ۴ مراحل مختلف مربوط به اين پروتكل آمده است. همانطور كه در شكل ۴ قسمت (الف) مي بينيم ، S_1 و S_2 قبل از ارسال داده broadcast بايد از نود BLAC تقاضاي يک برچسب كنند. اين كار را با ارسال Unicast پيغامهاي *Broadcast_Label_Request* انجام مي دهند. نود BLAC نيز همانطور كه در قسمت (ب) شكل نشان داده شده است توسط broadcast كردن پيغامهاي *Broadcast_Label_Assignment* به شيوه RPF در شبكه ، برچسبهاي L_1 و L_2 را به آنها اختصاص مي دهد. مثلاً LSR_9 را در نظر بگيريد. واسط مربوط به كوتاهترين فاصله از اين LSR تا نود BLAC مستقيماً آنرا به اين نود متصل مي كند و لذا iif_{RPF} براي اين LSR مي باشد. لذا پيغامهايي را كه برچسب X در بالاي پشته آنها است و از طريق اين واسط دريافت مي كند ، به دو واسط ديگر خود ارسال مي كند. اما همانطور كه ديده مي شود ، LSR_9 اين پيغامها را از طريق واسط ديگر خود كه به LSR_3 متصل مي باشد نيز دريافت مي كند ولي چون اين واسط با iif_{RPF} در آن متفاوت است ، آنها را دور مي اندازد. به اين شيوه ارسال RPF مي گويند. توجه كنيد كه چون در اين مثال شبكه متقارن در نظر گرفته شده است ، لذا كوتاهترين فاصله بين دو نود در جهت معكوس و مستقيم يكسان مي باشد ولي در شبكه هاي موجود ، اكثر مسيرها نامتقارن هستند [۳۱] و iif_{RPF} در LSR ها در مسير معكوس دريافت داده تعيين مي شود. در اين شكل پيكانهاي دوطرفه به معناي اين است كه پيغامها در هر دو جهت بر روي لينک مربوطه ارسال مي شوند. مثلاً در لينک بين LSR_4 و LSR_7 در يک جهت پيغامها توسط LSR_4 و در جهت ديگر توسط LSR_7 ارسال مي شوند.



شکل۴: مراحل مختلف پروتکل broadcast الف) تقاضاي برچسب broadcast توسط منابع ب) تخصيص برچسب broadcast توسط نود BLAC به شيوه RPF

پس از اينکه S_1 و S_2 پيغام *Broadcast_Label_Assignment* مربوطه را از نود BLAC دريافت کردند ، داده هاي broadcast خود را با برچسبهاي L_1 و L_2 بهمان صورت که قبلاً توضيح داده شد ارسال مي نمايند. وقتي که S_1 و S_2 ديگر داده اي براي ارسال نداشتند ، پيغام *Broadcast_Label_Release* را مانند قسمت (الف) شکل به نود BLAC ارسال مي کنند و بدنبال آن نود BLAC نيز پيغام *Broadcast_Label_Release* مربوطه را مانند قسمت (ب) شکل شماره ۴ به تمام شبکه broadcast مي نمايد.

اگر قرار باشد پيغامهاي *Broadcast_Label_Assignment* و *Broadcast_Label_Release* ارسالي توسط نود BLAC بازاي هر منبع جداگانه در شبکه broadcast شود ، علاوه بر اينکه پهناي باند اشغالي توسط اين پيغامها در شبکه زياد خواهد بود ، تعداد پيغامهايي که هر LSR بايد پردازش کند نيز بالا مي رود و لذا بار پردازشي زيادي را بر آنها تحميل مي نمايد. علاوه بر آن نود BLAC بار زيادي را براي ارسال جداگانه آنها خواهد داشت. براي رفع اين مطلب مي توان در نود BLAC به اندازه زمان $T_{Delayed\_Response}$ قبل از ارسال هر پيغام صبر کرد تا در صورتيکه در اين مدت پيغامهاي ديگري براي ارسال آماده شدند ، آنها را با هم در يک پيغام ترکيب کند و بجاي ارسال جداگانه آنها اين پيغام جديد را ارسال کند. چگونگي کد کردن اين پيغامهاي جداگانه در يک پيغام مشترک و تغييرات لازم در هر الگوريتم ارائه شده در حال بررسي مي باشد.

۶ پشتيباني از گروههاي multicast حالت مجتمع در MPLS

در اين بخش ما پروتکل ارائه شده قبلي را گسترش مي دهيم و يک پروتکل جديد براي پشتيباني از گروههاي multicast حالت مجتمع در MPLS ارائه مي کنيم. براي اينکار بايد به طريقي بين داده broadcast و multicast تفاوت گذاشته شود. لذا S بايد در هنگام درخواست يک برچسب از نود BLAC توسط پيغام *Broadcast_Label_Request* آدرس گروه multicast را که IP_G مي باشد در قسمت payload آن بياورد و نود BLAC نيز اين آدرس را در پيغام *Broadcast_Label_Assignment* خود به همين صورت اضافه کند. در اينصورت هر LSR با دريافت پيغام اخير يک entry جديد در جدول MFT (Multicast Forwarding Table) خواهد ساخت. هر entry اين جدول بصورت زير است :

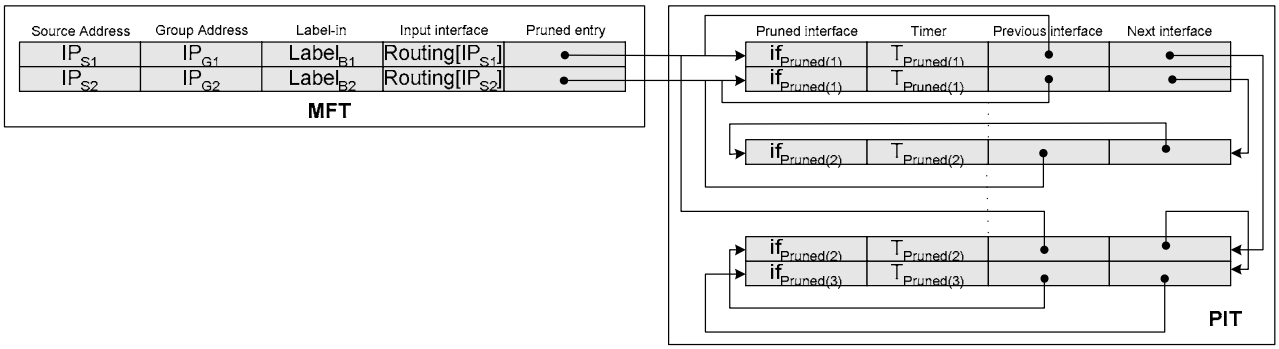
Source Address	Group Address	Label-in	Input interface	Pruned entry
IP_S	IP_G	$Label_B$	Routing[IP_S]	NULL

در اين جدول *Source Address* و *Group Address* با هم مشخص کننده درخت توزيع multicast می باشند که اين entry متعلق به آن است. برای اينکه طول entry های اين جدول ثابت باشد و بتوان آنرا براحتی در سخت افزار (مثلاً با استفاده از CAM) پياده سازی کرد ، بجای اينکه ليست واسطهای خروجی که پيغام حذف برای آنها آمده است را در هر entry نگهداری کنيم ، يک فيلد با طول ثابت *Pruned entry* را در هر entry اين جدول قرار داده ايم که يک اشاره گر به entry جدول ديگری بنام PIT (Pruned Interfaces Table) است که اولين واسط حذف شده برای اين درخت در آنجا ذخيره شده است.

اما سؤال اين است که چگونه جدول PIT ساخته می شود ؟ گفتيم که با دريافت پيغامهای تخصيص و رها کردن برچسب از نود BLAC يک entry به MFT اضافه می شود و يا از آن حذف می گردد. هنگاميکه S از برچسب اختصاص داده شده توسط BLAC اطلاع حاصل کرد ، اولاً آدرس IP_G را در فيلد آدرس مقصد داده multicast ارسالی خود قرار می دهد و ثانياً بجای برچسب Y که برای داده های broadcast استفاده می شد از برچسب ديگر رزرو شده Z استفاده می کند. بقيه قسمتهای داده مانند حالت broadcast می باشد. هنگاميکه يک روتر برگ (روتری که داده های S را به هيچ روتر ديگر پايين دستی ارسال نمی کند) داده multicast منبع S را دريافت کرد ، در صورتيکه بر اساس پروتکل IGMP [۳۴] متوجه شد که عضوی از گروه مورد نظر به آن متصل نيست ، يک پيغام prune را در لايه ۳ به سمت منبع S ارسال می کند. هر LSR در مسير رسيدن به S با دريافت اين پيغام ابتدا entry مربوط به (IP_S , IP_G) را در جدول MFT خود پيدا می کند و سپس واسطی که پيغام روی آن آمده است را در جدول PIT جستجو می کند. اگر اين واسط موجود بود زمانسنج آنرا برابر مقدار

اولیه T_{Prune} قرار می دهد و بدین ترتیب entry مربوطه را refresh می کند. اگر واسط را پیدا نکرد ، تغییرات لازم را در جدول PIT می دهد تا این واسط را بعنوان واسط جدید حذف شده در نظر بگیرد. اگر تمام واسطهای یک LSR از درخت مربوطه حذف شده بود ، LSR یک پیغام prune به LSR بعدی در مسیر رسیدن به S ارسال خواهد کرد. برای برقراری مجدد یک شاخه حذف شده از ارسال پیغامهای graft در لایه ۳ به شیوه مشابه استفاده می شود. به این ترتیب درختهای multicast در محیط MPLS ساخته خواهند شد.

جداول MFT و PIT مانند شکل ۵ به یکدیگر مربوط هستند.



شکل ۵: رابطه جداول PIT و MFT

همانطور که در شکل ۵ مشاهده می شود ، هر entry جدول PIT شامل یک واسط (*Pruned interface*) و یک زمانسنج (*Timer*) می باشد. با پایان پذیرفتن هر زمانسنج ، entry مربوطه از این جدول حذف می شود. هر واسطی که از یک درخت حذف می شود در اولین جای خالی جدول PIT می تواند قرار گیرد و اشاره گرهای *Previous interface* و *Next interface* در آن و واسط قبلی حذف شده در درخت باید طوری تنظیم شوند که این واسط جدید در انتهای لیست پیوندی واسطهای حذف شده از یک درخت قرار گیرد که سر آن توسط *Pruned entry* مربوطه مشخص می شود. فیلدهاي *Next interface* و *Previous interface* به ترتیب در آخرین واسط حذف شده و در اولین واسط حذف شده از یک درخت برابر آدرس entry خود آن واسط در جدول PIT قرار داده می شوند. این بدلیل نداشتن مقدار *NULL* در میان آدرسهاي موجود مي باشد و براى پايان دادن به ابتدا و انتهاي ليست پيوندي است. جدول PIT به راحتی در حافظه RAM پیاده سازی می شود.

خوبی این روش این است که جداول MFT و PIT از نظر فضاي اشغالي بهينه خواهند بود. علت این امر این است که چون گروه مجتمع فرض شده بنابراین تعداد واسطهایی که از هر درخت حذف می شود یا صفر است و یا مقدار کوچکی است و لذا جدول PIT آن کوچک خواهد بود.

هر LSR با دریافت یک داده با برچسب Z در اول پشته برچسبها می فهمد که داده multicast است و با جستجوی برچسب بعدی پشته در جدول MFT خود و مراجعه به جدول PIT و الگوریتمهای مشابه حالت broadcast ، درصورت لازم آنرا به *AllLinks – iif – AllPrunedLinks* ارسال خواهد کرد.

براي اجتناب از برچسب اضافي Z مي توانيم مانند مکانيزم broadcast عمل کنيم. ما از مقدار موجود در سرآيند لايه ۲ هنگاميکه وجود دارد براي تشخيص يک بسته multicast از بسته هاي unicast و broadcast استفاده مي کنيم. اين مقدار در حال حاضر براي PPP (مقدار هگز ۰۲۸۳ در فيلد *Protocol* براي MPLS multicast) ، Ethernet و IEEE 802.3 (مقدار هگز ۸۸۴۸ در فيلد *ethertype*) تعريف شده است[۲۹][۳۶]. ما پيشنهاد مي کنيم که در تکنولوژيهاي ديگر لايه ۲ که چنين مقاديري موجود نيست هم براي MPLS multicast مقادير خاصي تعريف شوند.

۷ مسائل جانبي

۱. ما در اين پروتکل سه برچسب خاص X ، Y و Z را معرفي کرديم که بايد يک مقدار رزرو شده را در حوزه MPLS داشته باشند. همانطور که در [۲۹] آمده است، مقادير برچسب ۴ تا ۱۵ در حال حاضر رزرو شده اند و بر اساس موافقت عمومي IETF ممکن است توسط IANA تخصيص داده شوند. لذا ما از اين محدوده براي پروتکل خود مقادير ۴ تا ۶ را براي اين برچسبهاي خاص پيشنهاد مي کنيم.

۲. از آنجا که پروتکلهای ارائه شده باعث مي شود که يک برچسب ۴ بايتي X ، Y يا Z به داده ها يا پيغامهاي مربوط به پروتکل اضافه شود ، لذا بر روي MTU محاسبه شده توسط روالهاي Path MTU Discovery تأثير مي گذارد و بنابراين بايد ۴ بايت از مقدار محاسبه شده کمتر در نظر گرفته شود.

۳. در پروتکلهای ارائه شده بعلت جدا بودن جدول ارسال داده هاي broadcast و multicast از جدول ارسال داده هاي Unicast و قرار دادن برچسب Y يا Z در اول پشته برچسب داده ها ، فضاي برچسبهاي broadcast ، multicast و Unicast از يکديگر جدا مي باشند و بنابراين مصرف شدن برچسبها در يکي روي ديگري تأثيري ندارد. از آنجا که بازاي هر منبع تنها يک برچسب broadcast يا multicast مصرف خواهد شد ،

لذا مي توان با استفاده از اين پروتكلها تعداد ۱۶ - $۲^{۲۰}$ منبع ارسال داده broadcast يا multicast را پشتيباني كرد (هر برچسب ۲۰ بيت است و مقادير برچسبهاي ۰ تا ۱۵ رزرو شده اند [۲۹]).

۴. براى اينكه نود BLAC در ميزان بار بالا نقطه failure نشود و توان پردازشى لازم را داشته باشد مى توان آنرا بصورت يک cluster محاسباتى طراحى كرد كه اتصال اجزاى محاسباتى در آن از طريق يک زيربناى تحمل پذير نسبت به خطا و بالانس كننده بار انجام شود. اين كار در پروتكلهاى multicast ديگر ارائه شده براى محيطهاى MPLS و IP نيز انجام گرفته است [۲۳][۳۰].

۵. در آنچه كه تاكنون ديديم پروتكل لايه ۳ IPv4 فرض شده است. اما هنگاميكه پروتكل لايه ۳ چيز ديگرى باشد لازم است تا آخرين برچسب موجود در پشته برچسبها يک برچسب خاص مربوط به آن پروتكل باشد [۲۹]. اين باعث خواهد شد تا در بعضى الگوريتمها تغييرات جزيى صورت بگيرد. ما در اين مقاله فرض كرده ايم كه اين برچسب وجود ندارد.

۶. پروتكلهاى ارائه شده در حد intra-domain قابل گسترش مى باشند. لذا در اين مقاله فرض شده است كه محيط MPLS داراى hierarchy نمى باشد. تغييرات لازم براى پشتيبانى از محيطهاى inter-domain با hierarchy در دست مطالعه است.

۷. در اين مقاله تنها استفاده از تكنيک RPF در پروتكلها توضيح داده شده است. تكنيكهاى بهينه تر RPB و Extended RPF نيز كه در [۲۷] ، [۲] و [۲۸] آمده اند با تغييرات جزيى در الگوريتمهاى ارائه شده قابل پياده سازى مى باشند.

۸. مسئله ديگرى كه در دست بررسى مى باشد تأثير خرابى در يک نقطه شبكه روى طرز كار پروتكلهاى ارائه شده مى باشد. مثلاً هنگاميكه بدليل خرابى يک لينک ، جواب درخواست يک منبع از نود BLAC به آن نمى رسد و درخواست تكرار مى شود ، آيا بايد جواب آن توسط نود BLAC دوباره broadcast شود يا اينكه بهتر است اينبار بصورت unicast جواب تنها براى خود منبع ارسال شود؟

۸ كارهاي ارائه شده قبلي

عمل سوييچ كردن برچسب IP multicast در ATM براى اولين بار در مدل اوليه اى از يک سوييچ روتر انجام شد كه از يک ماشين UNIX و يک سوييچ ATM از شركت Alcatel تشكيل شده بود [۳۵]. اين LSR قادر بود داده multicast را با استفاده از پروتكل PIM-SM [۴][۵][۱۲] در لايه IP و ارتباطات p2mp در ATM ارسال كند. در LSR آنها درخت ساخته شده در لايه ۳ به يک درخت p2mp در لايه ۲ نگاشته مى شد. براى انجام عمليات مورد نظر در طرح آنها به پشتيبانى از ارسال مختلط نياز بود. آنها تصميم گرفتند LSP ها را بر اساس تغييرات انجام گرفته در MFC (حافظه نهانى ارسال multicast) ايجاد كنند كه يک روش trigger كردن بر اساس ترافيک مى باشد [۱۶]. MFC يک زير مجموعه cache شده از MRT (جدول مسيريابى روتر) مى باشد. MRT هنگاميكه در مسيرها تغييرى ايجاد مى شود MFC را به روز مى كند. با آمدن يک بسته multicast اگر يک cache miss در MFC اتفاق افتاد ، entry مربوطه براى گروه multicast از MRT تقاضا مى شود. با وجود اينكه آنها در پياده سازى خود پروتكل PIM-SM را استفاده كرده اند ، روش آنها براى PIM-DM و DVMRP هم كار مى كند. اما ماهيت پخش و حذف انتخابى اين پروتكلها باعث مى شود تا با هر بار اتمام مدت زمان سنج و بطور متناوب cache miss داشته باشيم. اين امر روش ارائه شده را نا مناسب مى كند. بعلاوه آنها روش تخصيص برچسب خود را توضيح نمى دهند.

Ooms و همكاران در [۱۶] و RFC3353 [۱۷] چهارچوبى براى پياده سازى multicast در MPLS ارائه كرده اند و به بررسى مشكلات آن پرداخته اند. اما هيچ راه حل جامعى در خصوص چگونگى پياده سازى پيشنهادات خود ارائه نداده اند.

Farinacci و همكاران [۱۴] روش حمل قاچاقى (piggy-backing) را پيشنهاد نمودند. روش ارائه شده براى PIM-SM بدين صورت است كه برچسبها را بصورت قاچاقى همراه با پيغامهاى اين پروتكل مى فرستيم. لذا بايد تغييراتى در پيغامهاى اين پروتكل داده شود. اما روش آنها براى PIM-DM و بقيه پروتكلهاى مجتمع ضعيف و غير قابل قبول است زيرا در اين پروتكلها پيغام مناسبى وجود ندارد كه براى حمل قاچاقى استفاده شود. آنها پيشنهاد مى كنند كه پيغامهاى اضافه متناوبى به اين دسته از پروتكلها اضافه شود كه خود سربار اضافه اى است و كل مزيت روش را از بين مى برد. روش حمل قاچاقى داراى معايب متعددى است كه در [۱۶][۱۷] بتفصيل آورده شده است.

Acharya و همكاران [۱۵] پيشنهاد مى كنند كه به هر زوج (S, G) يک برچسب با آمدن ترافيک اختصاص داده شود. هنگاميكه يک LSR يک FEC جديد multicast را تشخيص داد براى تعيين واسطهاى خروجى آن از مسيريابى لايه ۳ استفاده مى كند. سپس بازاى هر واسط خروجى يک برچسب به شاخه مربوطه درخت اختصاص مى دهد. روش آنها براى گروههاى حالت مجتمع در مقايسه با پروتكل ما چند اشكال دارد. اولاً از آنجا كه آنها بازاى هر شاخه درخت multicast يک زوج واسط و برچسب را ذخيره مى كنند ، اندازه جدول MFT بزرگترى نسبت به ما دارند. در گروههاى مجتمع اكثر واسطها در بيشتر LSR ها شاخه اى از درخت مى باشند و اين باعث بزرگتر بودن جداول ارسال آنها نسبت به ما مى باشد چون ما شاخه هايى را كه عضو درخت نيستند نگه مى داريم. دوماً entry ها در جدول MFT آنها داراى طول متغير مى باشد و لذا پياده سازى سخت افزارى آن نسبت به جداول موجود در پروتكل ما سختتر است. سوماً در روش آنها تخصيص و توزيع برچسب در همه LSR ها انجام مى گيرد كه در ساخت درخت تأخير اضافه ايجاد مى كند. در پروتكل ما تخصيص برچسب توسط BLAC انجام مى گيرد و سپس به تمام LSR ها توزيع مى شود. هنگاميكه منبع از تخصيص برچسب آگاه شد مى تواند از آن براى ارسال داده هاى خود استفاده كند حال آنكه در بعضى قسمتهاى شبكه ممكن است هنوز اطلاعات مربوط به اين تخصيص نرسيده باشد. چهارم اينكه وقتى استخر برچسب (label pool)

بین تمام واسطها در یک LSR مشترک باشد در پروتکل ما با مصرف کردن تنها یک برچسب از فضای برچسب multicast ، برچسب کمتری نسبت به روش آنها مصرف می شود. در این حالت در روش آنها بازای هر شاخه درخت موجود در LSR یک برچسب مصرف می شود.

یک روش توزیع برچسب بر اساس ترافیک در [۲۶] ارائه شده و پروتکلي براي گروههاي مجتمع بر اساس آن پیشنهاد شده است. روش آنها همان بدیهای روش [۱۵] را نسبت به پروتکل ما دارد. اندازه جدول MFT در پروتکل آنها حتی از [۱۵] هم بزرگتر است زیرا آنها مجبور به تکرار اطلاعات مشترک بین entry های مربوط به یک درخت برای دستیابی به entry های با طول ثابت در جدول خود می باشند.

روش ما نسبت به هر دو روش [۱۵] و [۲۶] پهناي باند بیشتري مصرف مي کند. این امر چندین دلیل دارد. اول این که ممکن است داده برچسب خورده broadcast مربوط به یک منبع به یک LSR رسیده باشد و اطلاعات مربوط به تخصیص برچسب آن هنوز در راه باشد. در این حالت ما بصورت موقتی از روش غیر بهینه broadcast بدون چک کردن RPF استفاده می کنیم که پهنای باند بیشتری مصرف می کند. همانطور که گفتیم برای رفع این مطلب می توانیم با هزینه کردن میزان پردازش بیشتر در LSR ، داده ها را با استفاده از سرآیند لایه ۳ و بصورت بهینه ارسال کنیم. دوم اینکه ۴ بایت برچسب خاص اضافی که ممکن است در بالای پشته برچسبها در بسته ها قرار گیرد پهنای باند اضافی مصرف می کند. همانطور که گفتیم این امر می تواند با استفاده از پشتیبانی لایه ۲ مرتفع شود. سوم اینکه پیغامهاي ارسالي BLAC پهناي باند شبکه را مصرف مي کنند. همانطور که دیدیم اثر آنها را می توان با ترکیب چندین پیغام مجزا با هم کاهش داد.

در [۲۳] و [۲۴] یک پروتکل multicast جدید براي گروههای پراکنده در محیط MPLS ارائه شده است. در این پروتکل بازاي هر گروه یک نقطه مرکزي بنام NIMS (Network Information Manager System) در شبکه وجود دارد که کار آن محاسبه درخت multicast یک گروه با استفاده از جمع آوري تمام پیغاماي Join و Prune از اعضاي گروه مي باشد.

روش دیگري که در مقاله [۲۵] براي ارسال داده هاي multicast در یک محیط MPLS پیشنهاد شده است فرض مي کند که اعضاي یک گروه multicast در یک محیط MPLS تنها روي LER ها حضور دارند. با این فرض ، در پروتکل آنها یک درخت ساخته مي شود که نقاط انشعاب آن تنها در LER ها قرار دارند. بدین وسیله مي توانند داده هاي multicast را نیز مانند Unicast بصورت aggregate شده بین دو LER ارسال کنند. در این روش چند نکته وجود دارد که قابل بررسي است. اولاً صحت فرض انجام شده بر عدم وجود اعضاي گروههاي multicast در LSR هاي حوزه MPLS قابل تردید است. ثانیاً از آنجا که در کاربردهاي multicast معمولاً حجم داده زیادي منتقل مي شود ، لذا با افزایش تعداد گروههایي که داده خود را بین دو LER یکسان aggregate مي کنند ، ترافیک گذرنده از این مسیر مي تواند بسیار زیاد شود و یک تنگنا در قابلیت گسترش پروتکل باشد. علاوه بر این چون داده هاي multicast تنها از مسیرهاي ثابت بین LER ها مي توانند بگذرند ، utilization لینکهاي موجود در شبکه پایین مي آید و تنها به مسیرهاي ذکر شده محدود مي شود. بدي دیگر روش آنها این است که در هر LER چون درختهاي گروههاي مختلف ممکن است از یکدیگر جدا بشوند باید داده هاي multicast مربوط به هر درخت از درخت دیگر تفکیک شوند. اما براي این کار لازم است که از سرآیند لایه ۳ آنها استفاده شود و این بمعناي آن است که ارسال داده ها نمي تواند بصورت end-to-end با استفاده از سوییچ شدن برچسب انجام بگیرد و تأخیر ارسال داده های آن بیشتر می شود. بعلاوه این امر باعث افزایش بار پردازشی LER ها می شود. همچنین عملیات لایه ۳ مذکور روی داده های unicast نیز باید انجام شود زیرا ترافیک آنها قسمتی از ترافیک aggregate شده است. لذا روش آنها باعث ایجاد اختلال در سوییچ شدن برچسب داده های unicast می شود که مطلوب نیست. بدی دیگری که این روش در گروههای مجتمع دارد هدر دادن منابع شبکه است. چون اکثر LER ها در این گروهها عضو درخت می باشند ، با تشکیل LSP های جداگانه unicast بین آنها ، داده در جاهایی که این مسیرها هم پوشانی دارند از لینکهای مربوطه چند بار عبور خواهد کرد.

۹ نتیجه گیری و قدمهاي آینده

ما برای اولین بار در این مقاله پروتکلی برای broadcast در محیط MPLS بصورت بهینه ارائه کردیم. مکانیسم طراحي شده از یک نود مرکزي به اسم BLAC براي مدیریت تخصیص برچسبهاي broadcast در داخل حوزه MPLS استفاده مي کند. این مکانیسم معادل مکانیسم broadcast موجود در IP مي باشد و مي تواند مورد استفاده دیگر کاربردها و پروتکلهایي که به broadcast نیاز دارند نیز قرار گیرد. پروتکل ارائه شده هیچ برچسبی از فضای برچسب unicast مصرف نمی کند. با استفاده از پروتکل مذکور ، پروتکل دیگری براي پشتیباني از گروههاي مجتمع multicast ارائه شده است که مانند DVMRP می باشد. این پروتکل نسبت به پروتکلهایی که تاکنون پیشنهاد شده برتریهای زیادی دارد. ما برای هر درخت multicast تنها یک برچسب از فضای برچسب multicast مصرف می کنیم و فضای برچسب unicast را دست نخورده باقی می گذاریم. علاوه بر آن با ذخیره یک entry بازای هر واسط که شاخه ای از درخت نباشد جداول ارسال کوچکتری نسبت به روشهای دیگران داریم. در ادامه این کار ، ما قصد داریم با استفاده از شبیه سازي ، میزان کارایي پروتکلهاي خود را اندازه گیري کنیم و مقادیر بهینه پارامترهاي طراحي ارائه شده در مقاله را بدست بیاوریم. بعلاوه قصد داریم مسائلي مانند چگونگي برخورد مؤثر با خرابي و گسترش دادن پروتکلها بصورت hierarchical را که در بخش ۷ به آنها اشاره شد بررسي کنیم.

منابع

1. B.Davie, Y.Rekhter, "MPLS: Technology and Applications", Morgan Kaufmann, 2000
2. S.E.Deering, D.R.Cheriton, "Multicast Routing in Datagram Internetworks and Extended LANs",ACM Transactions on Computer Systems, Vol.8, No.2, May.1990
3. A.J.Ballardie, "A New Approach to Multicast Communication in a Datagram Internetwork", Ph.D. Thesis, University of London, 1995

27. Y.K. Dalal, R. M. Metcalfe, “Reverse Path Forwarding of Broadcast Packets”, Communications of the ACM, Vol. 21, No. 12, Dec. 1978
28. A. J. Frank, L. D. Wittie, A. J. Bernstein, “Multicast Communication on Network Computers”, IEEE Software, Vol. 2, No. 3, pp. 49-61, May 1985
29. E. Rosen, D. Tappan, G. Fedorkow, Y. Rekhter, D. Farinacci, T. Li, A. Conta, “MPLS Label Stack Encoding”, RFC 3032, Jan. 2001
30. S. Keshav, S. Paul, “Centralized Multicast”, Proceedings of IEEE ICNP, 1999
31. V. Paxson, “End-to-End Routing Behavior in the Internet”, Proceedings of SIGCOMM ‘96
32. M.R. Macedonia, D.P. Brutzman, “MBone Provides Audio and Video Across the Internet”, IEEE Computer Magazine, Vol. 27, No. 4 , pp. 30 -36, April 1994
33. L. L. Peterson, B. S. Davie, "Computer Networks: A Systems Approach", Morgan Kaufmann Publishers, 2nd Edition, 2000
34. W. Fenner , “Internet Group Management Protocol, Version 2”, RFC 2236, Nov. 1997
35. P. Dumortier, D. Ooms, W. Livens, I. Girard, M. Ramalho, “IP Multicast Shortcut over ATM: A Winner Combination”, IEEE Globecom 1998, Nov. 1998
36. "Protocol Numbers and Assignment Services", http://www.iana.org/numbers.html

4. S.Deering, D.Estrin, D.Faranacci, V.Jacobson, C.G.Liu, L.Wei, "An Architecture for Wide-Area Multicast Routing", Proc. SIGCOMM'94, Computer Communication Review, Vol.24, No.4, Oct.1994
5. S.Deering, D.Estrin, D.Faranacci, V.Jacobson, C.G.Liu, L.Wei, "The PIM Architecture for Wide-Area Multicast Routing", IEEE/ACM Transactions on Networking, Vol.4, No.2, April 1996
6. T.Ballardie, P.Francis, J.Crowcroft, "Core Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing", ACM SIGCOMM'93
7. K.C.Almeroth, "The Evolution of Multicast: From the MBone to Inter-Domain Multicast to Internet2 Deployment", IEEE Network, Jan./Feb. 2000
8. S.Casner, S.Deering, “First IETF Internet Audiocast”, ACM SIGCOMM, Computer Communications Review, Vol. 22, No. 3, July 1992
9. T.Billhartz, J.B.Cain, E.Farrey-Goudreau, D.Fieg, S.Batsell, "Performance and Resourse Cost Comparisons for the CBT and PIM Multicast Routing Protocols", IEEE Journal on Selected Areas in Communications, April 1997
10. L.Wei, D.Estrin, "Multicast Routing in Dense and Sparse Modes: Simulation study of Tradeoffs and Dynamics", University of Southern California, Computer Science, Technical Report, 1995
11. L.Wei, D.Estrin, "The Trade-offs of Multicast Trees and Algorithms", in Proc. International Conference on Computer Communications and Networks, Sep. 1994
12. B.Fenner, M.Handley, H.Holbrook, I.Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", IETF Draft, draft-ietf-pim-sm-v2-new-03.ps, July 2001
13. T.Maufer, C.Semeria, "Introduction to IP Multicast Routing", IETF Draft, draft-ietf-mboned-intro-multicast-03.txt, July 1997
14. D.Farinacci, Y.Rekhter, E.C.Rosen, T.Qian, "Using PIM to Distribute MPLS Labels for Multicast Routes",IETF Draft, draft-faranacci-mpls-multicast-03.txt, Nov. 2000
15. A.Acharya, F.Griffoul, F.Ansari, "IP Multicast Support in MPLS", IEEE Proceedings on ATM Workshop, 1999
16. D.Ooms, W.Livens, "IP Multicast in MPLS Networks", Proceedings of the IEEE Conference on High Performance Switching and Routing, 2000
17. D.Ooms, B.Sales, W.Livens, A.Acharya, F.Griffoul, F.Ansari, "Overview of IP Multicast in a Multi-Protocol Label Switching (MPLS) Environment", RFC 3353, Aug.2002
18. D.Waitzman, C.Partridge, S.Deering, "Distance Vector Multicast Routing Protocol", RFC 1075, Nov.1988
19. E.Rosen, A.Viswanathan, R.Callon, "Multiprotocol Label Switching Architecture", RFC 3031, Jan.2001
20. A.Viswanathan, N.Feldman, Z.Wang, R.Callon, “Evolution of Multiprotocol Label Switching”, IEEE Communications Magazine, May 1998
21. A.Watson, M.Sasse, “The Good, the Bad, and the Muffled : the Impact of Different Degradations on Internet Speech”, Proceedings of the 8th ACM international multimedia conference, Oct. 2000
22. G. Liu, X. Lin, “MPLS Performance Evaluation in Backbone Network”, IEEE International Conference on Communications (ICC 2002), Vol. 2, pp. 1179 – 1183, May 2002
23. A. Boudani, B. Cousin, “A New Approach to Construct Multicast Trees in MPLS Networks”, Proceedings of the Seventh International Symposium on Computers and Communications (ISCC 2002), pp. 913 – 919, July 2002
24. A. Boudani, B. Cousin, “An Effective Solution for Multicast Scalability: The MPLS Multicast Tree (MMT)”, work in progress, http://www.ietf.org/internet-drafts/draft-boudani-mpls-multicast-tree-02.txt
25. B. Yang and P. Mohapatra, “Edge Router Multicasting with MPLS Traffic Engineering”, IEEE International Conference on Networks (ICON 2002), Aug. 2002
26. Z. Zhang, K. Long, W. Wang, S. Cheng, “The new mechanism for MPLS supporting IP multicast”, The 2000 IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS 2000)