به نام خداوند بخشنده‌ی مهربان

تمرین برنامه‌نویسی سوم[1]
شبکه‌های کامپیوتری

مدرس: مهدی خرازی **بهار ۱۳۹۱**

در این تمرین شما باید یک مسیریاب ساده پیاده‌سازی کنید. این مسیریاب از پروتکل OSPF پشتیبانی خواهد کرد.

۱ مقدمه

در تمرین قبل، بخش‌های اولیه مرتبط با یک مسیریاب ساده را که مبتنی بر پروتکل OSPF کار می‌کند، ایجاد کردیم. این بخش‌ها عبارت بودند از پروتکل HELLO و انتخاب Designated Router و Backup Designated Router. در این تمرین بخش دیگری از این پروتکل با عنوان Flooding را پیاده‌سازی خواهید نمود.

این مستند، شامل خلاصه‌ای از جزئیاتی که باید پیاده‌سازی شوند است. اما برای کسب اطلاعات تکمیلی، مطالعه‌ی منابع زیر مفید است. در بخش‌هایی که برای پیاده‌سازی نیاز به مطالعه‌ی قسمتی خاص از یک منبع دارند، آدرس دقیق‌تر آورده شده است.

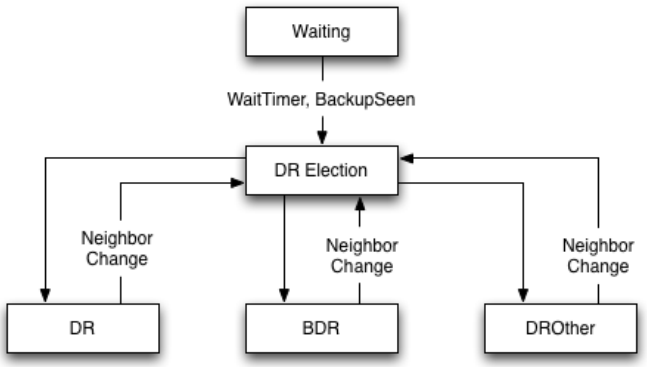
1- http://tools.ietf.org/html/rfc2328
2- http://fengnet.com/book/OSPFandISIS/ch02.html
3- http://en.wikipedia.org/wiki/Open_Shortest_Path_First
4- http://en.wikipedia.org/wiki/Link-state_advertisement

در این تمرین، فرض می‌شود که همه‌ی مسیریاب‌ها در یک ناحیه قرار دارند. بستر پیاده‌سازی تمرین، شبکه‌ی Ethernet است که از شبکه‌های با قابلیت broadcast محسوب می‌شود.

[1] با تشکر از بهنام مومنی، مهدی احمدی‌نژاد، کامیار اللهوردی، سجاد فولادی، علی‌محمد ربانی و مهرداد مرادی

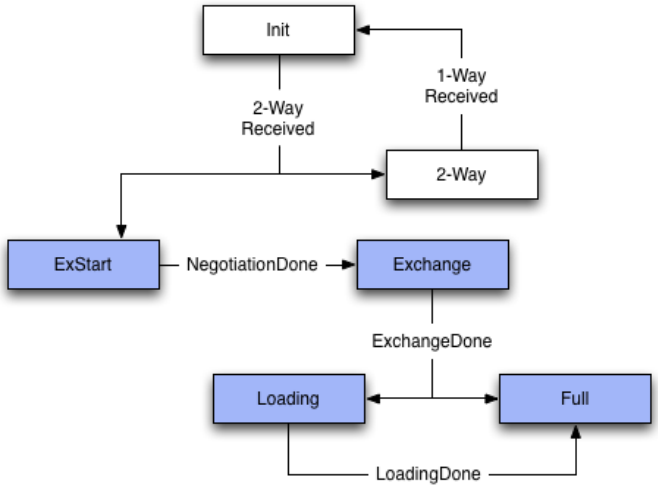
۲ نمای کلی

در تمرین قبل، کاری که انجام شد، معادل با تولید دو ماشین حالت بود. یکی برای هر اینترفیس از یک مسیریاب و دیگری برای هر یک از همسایگان آن. هر اینترفیس، وضعیت فعلی خود در شبکه را با استفاده از این ماشین حالت می‌داند و به ازای رخدادهای[۲] متفاوت در هر حالت، تصمیم‌گیری می‌کند. در تمرین قبل، این ماشین حالت به‌طور کامل و به شکل زیر پیاده‌سازی شد.



تصویر شماره ۱ – ماشین حالت مربوط به هریک از اینترفیس‌های مسیریاب

هر همسایه از هر اینترفیس یک مسیریاب نیز یک ماشین حالت دارد که بخشی از آن در تمرین قبل، و بخش دیگر آن در این تمرین پیاده‌سازی خواهد شد. این ماشین حالت به شکل زیر است. قسمت‌های رنگی نشان‌دهنده‌ی بخشهایی هستند که در این تمرین باید پیاده‌سازی شوند.



تصویر شماره ۲ – ماشین حالت مربوط به هر یک از همسایگان یک مسیریاب

[2] Event

۳ همجواری

برای اینکه مسیریاب‌ها بتوانند جداول مسیریابی خود را کامل کنند، نیاز دارند تا اطلاعات مربوط به توپولوژی و هزینه‌ی مسیرها را در اختیار داشته باشند. این کار از طریق تبادل این اطلاعات بین مسیریاب‌ها انجام می‌شود. در حالت عادی، هر مسیریاب باید اطلاعات خود را برای تمام بقیه‌ی مسیریاب‌ها ارسال کند تا همه‌ی مسیریاب‌های موجود در شبکه اطلاعات لازم را برای تکمیل جداول مسیردهی داشته باشند. در این حالت، در صورتی که تعداد مسیریاب‌های موجود در شبکه از حدی بیشتر باشد، تعداد این ارسال‌ها و دریافت‌ها بسیار بالا خواهد رفت و شبکه کارایی خود را از دست می‌دهد.

برای رفع این مشکل، در پروتکل OSPF، در هر شبکه، یک مسیریاب به عنوان Designated Router (و یک مسیریاب هم به عنوان Backup Designated Router) انتخاب می‌شود و هر مسیریاب موجود در شبکه تنها اطلاعات خود را با این مسیریاب همگام می‌کند. در نهایت Designated Router با تجمیع این اطلاعات، نتیجه را برای سایرین ارسال می‌کند. در واقع، بعد از انتخاب DR، این مسیریاب با تمام مسیریاب‌های دیگر شبکه مجاور[۳] می‌شود و تمام این مسیریاب‌ها اطلاعات خود را تنها با آن همگام می‌نمایند. BDR نیز با تمام مسیریاب‌های شبکه همجوار می‌شود.

به دلیل اینکه ایجاد بسته‌هایی با محتوای یکسان برای تک‌تک همسایه‌های مجاور به‌طور جداگانه امری بهینه نیست، DR بسته‌هایی را که می‌خواهد به سایر مسیریاب‌ها ارسال کند، به آدرس AllSPFRouters ارسال می‌نماید. سایر مسیریاب‌ها هم ازطریق آدرس AllDRouters با DR ارتباط خواهند داشت.. توجه داشته باشید، اگر مسیریابی که DR یا BDR نیست، بسته‌ای با مقصد AllDRouters دریافت کرد، باید آن را drop کند. جزئیات مربوط به مقصد هریک از بسته‌ها در ادامه و با توضیح هر بخش خواهد آمد.

جدول ۱ ــ مقصد انواع بسته‌های مورد استفاده در پروتکل OSPF

نوع بسته	مبدأ	مقصد
Database Description	*	Unicast (Directly to the neighbour)
LS Request	*	Unicast
LS Update / LS Acknowledge	DR or BDR	AllSPFRouters
	DROther	AllDRouters
	Retransmission	Unicast

همان‌طور که در تمرین قبل گفته شد، برای کم‌اثر کردن خارج شدن DR از شبکه، یک مسیریاب به عنوان مسیریاب برگزیده‌ی پشتیبان (BDR) انتخاب می‌گردد. مسیریاب‌هایی که نه DR هستند و نه BDR، رابطه‌ی مجاورت را با هر دوی این مسیریاب‌ها برقرار می‌کنند. BDR هم مانند سایر مسیریاب‌ها اطلاعات خود را با DR همگام می‌نماید. دقت کنید که BDR همانند DR

[3] Adjacent

بسته‌هایی را که به مقصد AllDRouters باشد، دریافت می‌کند؛ زمانی که DR به هر دلیلی از شبکه خارج شد، BDR جای آن را خواهد گرفت و به DR جدید شبکه تبدیل خواهد شد؛ که در تمرین پیش این مورد را پیاده‌سازی کرده‌اید.

همجواری و برقراری ارتباط مجاورت بین دو مسیریاب، فرآیند پیچیده‌ای نیست. برای این‌کار تنها کافی است که با رخداد 2-Way Received، به جای انتقال همسایه از حالت Init به 2-Way، آن را به ExStart منتقل کنیم. برای فهم بهتر این موضوع به تصویر شماره‌ی ۲ مراجعه کنید.

۴ Flooding

پس از آنکه مسیریاب‌ها با یکدیگر ارتباط همجواری برقرار کردند، برای تکمیل و یکسان کردن Database، فرآیند Flooding صورت می‌گیرد. در Flooding هر مسیریاب اطلاعاتی را که دارد به مسیریاب‌های همجوار خود می‌دهد. اگر یک مسیریاب بفهمد که همسایه‌اش اطلاعاتی دارد که خودش ندارد و یا اطلاعات به‌روزتری دارد، آنها را از او می‌گیرد.

فرآیندی که طی آن یک مسیریاب می‌فهمد تعدادی از LSAهای همسایه‌ی مجاورش را ندارد، Synchronization نام دارد. این فرآیند با استفاده از بسته‌های Database Description انجام می‌شود. Synchronization با توجه به تصویر شماره‌ی ۲، زمانی آغاز می‌شود که یک همسایه وارد حالت ExStart شود. با ورود همسایه به حالت Loading، فرآیند Flooding ادامه پیدا کرده و همسایه‌های مجاور، LSAهای مورد نیاز خود را دریافت می‌کنند.

۴.۱ LSA

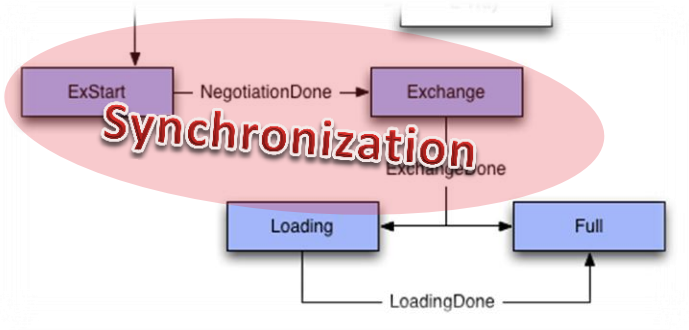
هر مسیریابی که در یک AS قرار دارد، یک یا چند Link-State Advertisement یا به اختصار LSA تولید می‌کند. به مجموعه‌ی این LSAهای تولید شده، Link-State Database گفته می‌شود. LSAها دارای انواع مختلفی هستند که در این تمرین تنها دو نوع Router-LSA و Network-LSA مورد استفاده قرار می‌گیرند. این Network-LSAها و Router-LSAها در کنار یکدیگر به طور کامل تعیین می‌کنند که مسیریاب‌ها در یک ناحیه چگونه به یکدیگر متصل شده‌اند. در پروتکل OSPF با یکسان سازی Link-State Database بین تمام مسیریاب‌های ناحیه، از طریق تبادل این LSAها، این امکان برای مسیریاب‌ها فراهم می‌شود که با داشتن توپولوژی کامل شبکه، جداول مسیریابی خود را ایجاد نمایند.

Router-LSAها توسط تمام مسیریاب‌های موجود در ناحیه تولید می‌شوند و شامل وضعیت تمام اینترفیس‌های مسیریاب تولید کننده‌ی LSA در ناحیه است. Network-LSAها نیز توسط DRها برای هر شبکه تولید می‌شود و شامل لیست مسیریاب‌های متصل به آن شبکه است.

۴.۲ Synchronization

زمانی که یکی از همسایه‌های مسیریاب وارد حالت ExStart شد (دقت کنید که هر مسیریاب تنها با مسیریاب‌های مجاور خودش وارد ExStart می‌شود؛ نه با هر همسایه‌ای)، در واقع رابطه‌ی مجاورت بین آن همسایه و مسیریاب برقرار شده است. این نقطه، نقطه‌ی آغاز عملیات Synchronization است. در پایان این فرآیند، هر یک از مسیریاب‌ها لیستی از LSAهای مورد نیاز خود تولید کرده است که بعداً در ادامه‌ی Flooding آنها را از این همسایه دریافت می‌کند.

فرآیند Synchronization با دریافت سرآیند تمامی LSAهای موجود در همسایه و تغییر حالت آن همسایه از Exchange به Loading یا Full به پایان می‌رسد. اگر در انتهای Synchronization، لیست LSAهای مورد نیاز مسیریاب خالی بود، حالت بعدی همسایه‌اش Full و در غیر این صورت Loading است.



تصویر شماره ۳ – انطباق Synchronization با ماشین حالت همسایه

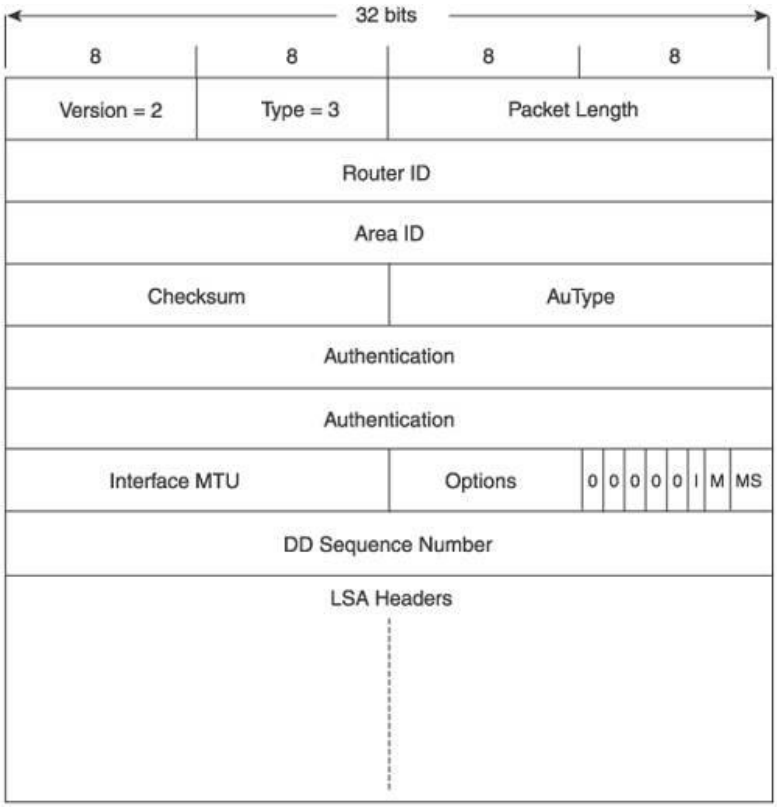
به طور کلی می‌توان عملیات Synchronization را در سه مرحله کلی خلاصه کرد:

۱. یکی از همسایه‌ها به عنوان Master و دیگری به عنوان Slave انتخاب می‌شود. مسیریاب Master ادامه‌ی فرآیند Synchronization را کنترل خواهد کرد. با انتخاب Master و Slave، رخداد NegotiationDone اتفاق می‌افتد و همسایه به حالت Exchange منتقل می‌شود.

۲. هر همسایه سرآیند تمام LSAهایی را که در پایگاه داده‌ی خود دارد برای دیگری ارسال می‌کند. برای این کار Master بسته‌های Database Descrition (DD) را ارسال کرده و Slave در همان بسته‌ای که با آن می‌خواهد بسته‌ی دریافتی را Acknowledge کند، سرآیند LSAهای خود را ارسال می‌کند.

۳. هر مسیریاب در این فرآیند، سرآیند LSAهای موجود در بسته‌های DD دریافتی را آزمایش کرده و درصورتی که LSA مربوط به آن سرآیند را نداشته باشد و یا نسخه‌ای قدیمی‌تر را داشته باشد، آن سرآیند را در لیست LSAهای مورد نیاز اضافه می‌کند.

۴.۲.۱ فرآیند انتخاب Master/Slave



تصویر شماره ۴ ـ بسته‌ی DD

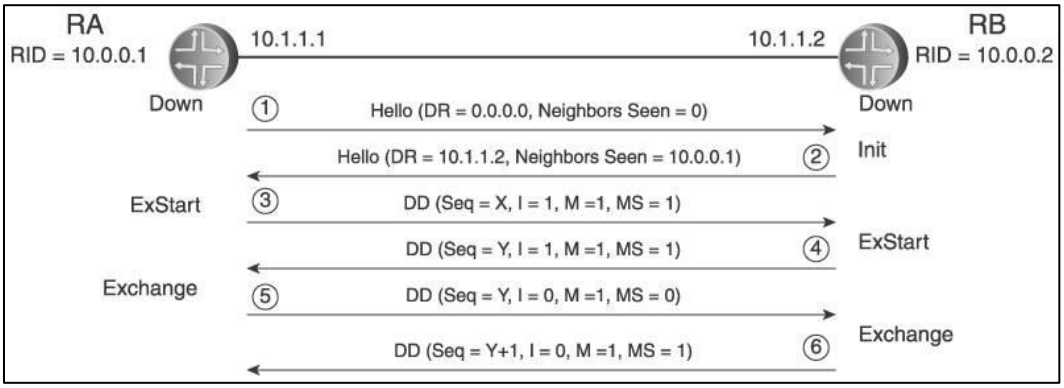
اولین قدم در اجرای Synchronization، انتخاب مسیریاب‌های Master و Slave است. انتخاب آنها از طریق ارسال بسته‌های DD با شرایطی خاص صورت می‌گیرد و با رخداد NegotiationDone به پایان می‌رسد. این شرایط به‌صورت زیر است:

۱- وقتی که همسایه وارد حالت ExStart شد، تا زمانی که از آن حالت به حالت بعدی منتقل نشده است، مسیریاب بسته‌های خالی DD را که بیت‌های I، M و MS[۴] آن برابر ۱ قرار داده شده‌اند، به همسایه ارسال می‌کند. قبل از ارسال این بسته یک Sequence Number (SN) برای همسایه به صورت تصادفی انتخاب می‌شود. در بسته‌ی DD، این عدد قرار می‌گیرد.

۲- مسیریاب گیرنده، که فرستنده را در وضعیت ExStart می‌بیند، بسته‌ی DD را دریافت می‌کند. اگر بیت‌های I، M و MS این بسته برابر ۱ بود، محتویات آن خالی بود و Router ID فرستنده بیش از Router ID خود بود، گیرنده Slave خواهد شد. این امر با ارسال یک بسته‌ی DD خالی با بیت‌های I و MS برابر با صفر اعلام می‌شود. SN بسته‌ی دریافتی به عنوان مقدار SN این همسایه ذخیره شده و برای ادامه‌ی فرآیند Synchronization مورد استفاده قرار می‌گیرد.

۳- اگر بیت‌های I و MS در بسته‌ی دریافتی برابر صفر بودند، SN بسته‌ی DD با SN در نظر گرفته شده برای آن همسایه برابر بود و Router ID همسایه از Router ID خود کوچکتر بود، همسایه خود را به عنوان Slave اعلام کرده و دریافت‌کننده‌ی بسته Master خواهد بود.

[4] I: Initial, M: More, MS: Master



تصویر شماره ۵ – فرآیند انتخاب Master/Slave

۴.۲.۲ Database Exchange

با ورود دو همسایه به حالت Exchange، فرآیند ارسال سرآیندهای LSA توسط بسته‌های DD، آغاز می‌شود. برای این کار ابتدا مسیریابی که به عنوان Master انتخاب شده است، SN را یکی زیاد کرده و بسته‌ای شامل سرآیند تمامی LSAهای خود ارسال می‌کند.

در طرف دیگر، Slave با دریافت یک بسته‌ی DD از Master، سرآیند LSAهای موجود در آن را بررسی کرده و LSAهای مورد نیاز خود را به لیست اضافه می‌کند. سپس یک بسته‌ی DD شامل سرآیند تمامی LSAهای خود که در آن SN برابر با SN در بسته‌ی دریافتی است، ارسال می‌کند.

دقت کنید که اندازه‌ی بسته‌های DD نباید از IP MTU تجاوز کند. در صورتی که تعداد LSAها به حدی باشد که در یک بسته قرار نگیرد، باید در چند بسته‌ی DD ارسال شود. برای سادگی، می‌توانید فرض کنید که تمام سرآیندهای LSAهای یک Router در یک بسته‌ی DD جا می‌شوند.

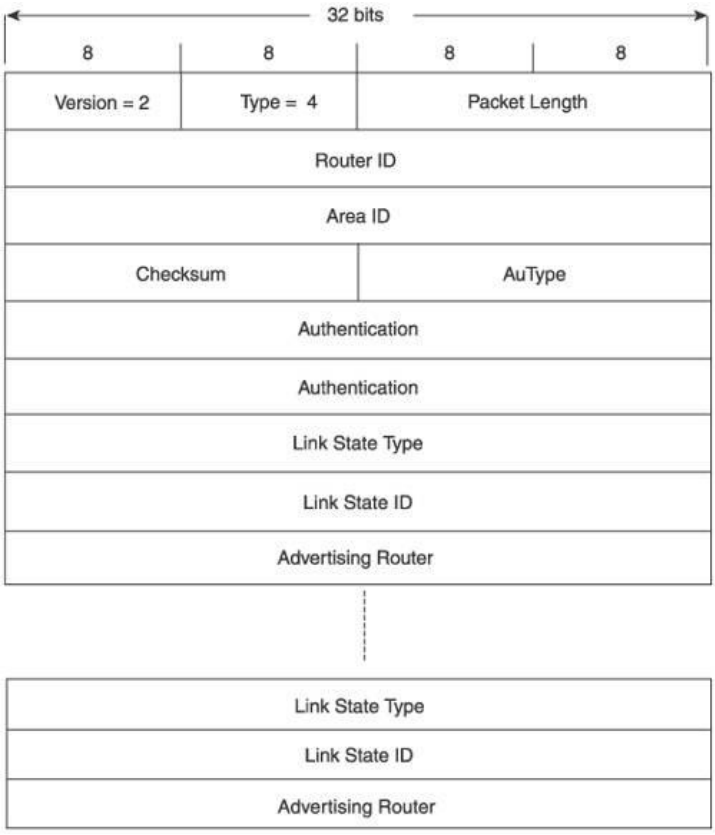
با آزمایش کردن سرآیندهای LSA، یک LSA به لیست LSAهای مورد نیاز اضافه می‌شود اگر سرآیند LSA دریافتی در بین سرآیند LSAهای موجود، وجود نداشته باشد. عدم وجود این سرآیند می‌تواند بدین معنی باشد که نسخه‌ای متفاوت با سرآیند دریافتی در مسیریاب وجود دارد و فقط به معنی عدم وجود کامل این سرآیند نیست.

پس از پایان یافتن عملیات Exchange، که با دریافت بسته‌ی DD که در آن بیت M برابر با صفر است، حالت همسایه به Loading یا Full تغییر می‌کند. اگر لیست LSAهای مورد نیاز خالی بود، حالت جدید Full و در غیر اینصورت Loading خواهد شد.

۴.۳ وضعیت Loading

با ارسال و دریافت بسته‌های DD که در آنها فقط سرآیند LSAها موجود است، مسیریاب‌ها می‌فهمند کدام LSAها را ندارند و یا نسخه‌ای قدیمی از آن را دارند و به این ترتیب لیستی از LSAهای موردنیاز خود تشکیل می‌دهند. بعد از پایان مرحله‌ی Exchange، همان‌طور که در قبل نیز اشاره شد؛ در صورتی که لیست LSAهای موردنیاز خالی نباشد، وارد وضعیت Loading می‌شویم. در این وضعیت مسیریاب یک بسته از نوع Link State Request ایجاد می‌کند و در آن لیست LSAهایی را که نیاز دارد قرار می‌دهد. همسایه‌ی دریافت کننده‌ی این درخواست، با استفاده از بسته‌های Link State Update، همه یا بخشی از LSAهای درخواستی را در اختیار مسیریاب درخواست کننده قرار می‌دهد. مسیریاب درخواست کننده، LSAهای دریافتی را از لیست LSAهای موردنیاز خود حذف می‌کند. در صورتی که لیست این LSAها خالی شد، به وضعیت Full خواهد رفت؛ در غیر این

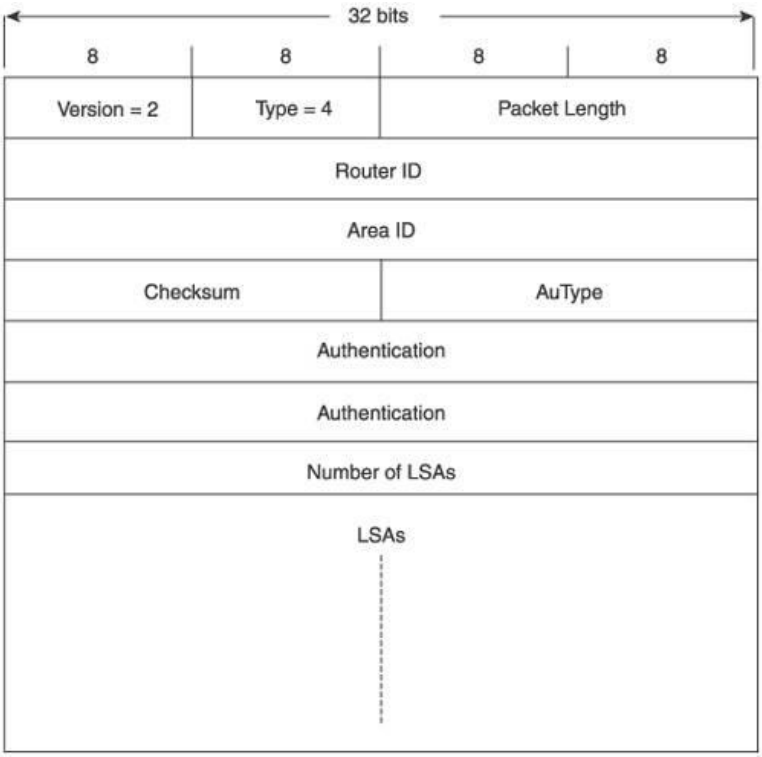
صورت، این عملیات درخواست تا زمان خالی شدن لیست ادامه پیدا می‌کند. می‌توانید فرض کنید که تمام LSAهای موردنیاز در یک بسته قابل درخواست هستند.



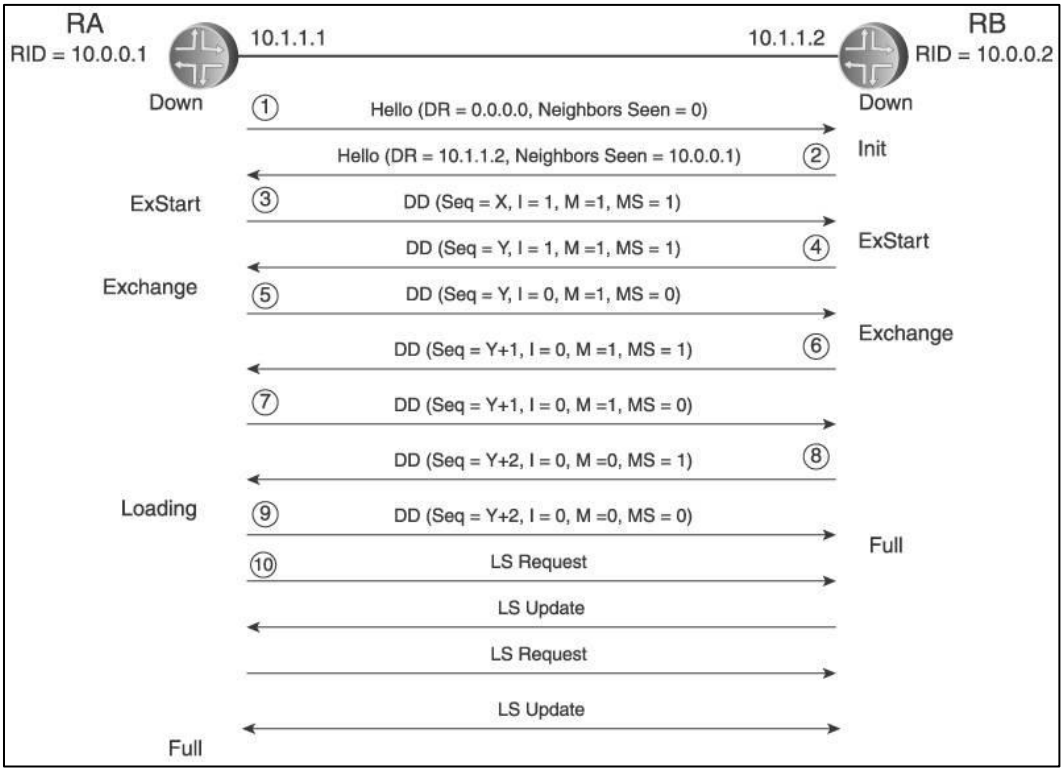
تصویر شماره ۶ ـ بسته‌ی LS Request

ممکن است برخی از LSAها در بسته‌های LS Request درخواست شده باشند ولی برای آن‌ها پاسخی دریافت نشده باشد (برای مثال همسایه فاقد آن LSA درخواستی باشد). این LSAها در بسته‌های درخواست جدیدی باید قرار داده شوند و بعد از طی شدن زمان RxmtInterval دوباره ارسال شوند. دقت کنید که در هر لحظه تنها باید یک بسته‌ی بدون جواب LS Request وجود داشته باشد.

دریافت کننده‌ی LS Request تنها زمانی به درخواست پاسخ می‌دهد که همسایه‌ی آن در یکی از وضعیت‌های Exchange، Loading و یا Full قرار داشته باشد. دقت کنید که در این مرحله، در وضعیت Loading، بسته‌های LS Update در لیست retransmission قرار نمی‌گیرند؛ در صورتی که جواب به گیرنده نرسد، گیرنده بعد از طی شدن زمان RxmtInterval دوباره اقدام به ارسال LS Request خواهد نمود. در مراحل آینده خواهید دید که بسته‌های LS Updateی که توسط مسیریاب‌ها تولید می‌شوند، در لیست Retransmission قرار می‌گیرند. به تفاوت این دو وضعیت توجه کنید.



تصویر شماره ۷ ـ بسته‌ی **LS Update**

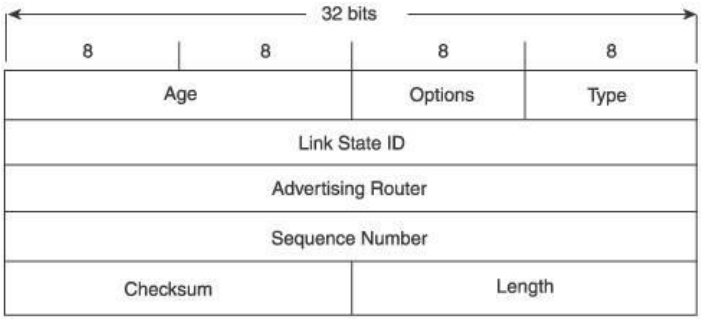


هنگامی که لیست LSAهای موردنیاز خالی شد، همسایه از وضعیت Loading به وضعیت Full خواهد رفت و مجاورت کامل شکل خواهد گرفت.

۴.۴ فرآیند Flooding

بسته‌های LS Update امکان اجرای فرآیند Flooding را فراهم می‌کنند. هر بسته‌ی LS Update می‌تواند شامل چندین LSA متفاوت باشد. برای اینکه این فرآیند قابل اطمینان باشد، هر LSA دریافتی باید به صورت جداگانه Acknowledge شود (به وسیله‌ی بسته‌های Link State Acknowledge). چند Acknowledge می‌توانند در یک بسته جمع شده و با هم ارسال شوند. برای تضمین رسیدن بسته‌ها، همان موقع ارسال بسته‌های LS Update، LSAهای آن در یک لیست Retransmission قرار خواهند گرفت. تا زمانی که Acknowledge برای یک LSA نیاید، در این لیست باقی خواهد ماند و در فواصل زمانی منظم عملیات Retransmit برای LSAهای موجود در این لیست انجام خواهد شد.

با دریافت یک بسته‌ی Link State Update، فرآیند Flooding آغاز می‌شود. در صورتی که بسته‌ی LS Update از همسایه‌ای دریافت شود که این همسایه در وضعیتی پیش از Exchange قرار دارد، باید بسته drop شود.



تصویر شماره ۸ ــ سرآیند LSA

هنگامی که یک بسته‌ی LS Update دریافت شد، به ازای هرکدام از LSAهای موجود در این بسته باید مراحل زیر صورت گیرد (دقت کنید که در اینجا منظور از همسایگی، در حقیقت رابطه‌ی همجواری است):

۱. فیلد LS Checksum باید کنترل شود و در صورت اشتباه بودن Checksum دیگر این LSA بررسی نمی‌شود.
۲. اگر مقدار فیلد LS Age برابر با MaxAge بود و هیچ موردی از این LSA در Link State Database مسیریاب نبود و هیچ‌کدام از همسایه‌های مسیریاب در وضعیت Loading یا Exchange قرار نداشتند، مراحل زیر انجام می‌شود:
   a. با ارسال بسته‌ی LS Acknowledge به همسایه ارسال کننده‌ی LSA دریافت آن را تایید می‌کند.
   b. بررسی LSA را متوقف می‌کند و به سراغ LSA بعدی می‌رود.
۳. در غیر این صورت، مورد مربوط به این LSA در Link State Database یافت می‌شود. در صورتی که این مورد متناظر در Database یافت نشود یا LSA دریافتی جدیدتر از آن باشد (شرایط جدیدتر بودن یک LSA در ادامه آمده است)، مراحل زیر باید انجام شود:
   a. در صورتی که هم‌اکنون یک مورد متناظر با این LSA در Database قرار دارد و این کپی از LSA از طریق فرآیند Flooding و در فاصله‌ی زمانی کمتر از MinLSArrival دریافت شده است، بررسی آن متوقف شده و باید LSA بعدی بررسی شود. در این حالت نیازی به Acknowledge نیز نمی‌باشد.
   b. در غیر این صورت، باید این LSA بلافاصله در برخی از اینترفیس‌های مسیریاب flood شود. جزئیات این عملیات در ادامه خواهد آمد.

c. این کپی از LSA از لیست Retransmissionها حذف می‌شود (یعنی اگر پیش از این به هر دلیلی این کپی از LSA در لیست ارسال مجدد قرار داشته است، به دلیل انجام flooding نیازی به ارسال مجدد آن نیست).
d. LSA جدید باید جایگزین قبلی در Database شود. همچنین باید زمان جاری به این LSA الصاق شود. دقت کنید که تا زمانی که حداقل به اندازه‌ی MinLSArrival از دریافت شدن این LSA نگذشته باشد، در طی این فرآیند این LSA نباید جایگزین گردد.
e. دریافت این LSA به کمک بسته‌های LS Acknowledge تأیید می‌گردد.

۴. در صورتی که نسخه‌ی کپی موجود در Database با LSA دریافتی یکسان بود، دو گام زیر صورت می‌گیرد:
a. در صورتی که این LSA در لیست Retransmission قرار داشت، مسیریاب منتظر Acknowledge برای این LSA است؛ در این حالت مسیریاب باید دریافت این LSA را به عنوان Acknowledge تلقی کرده و آن را از لیست Retransmission حذف کند.
b. دریافت این LSA به کمک بسته‌های LS Acknowledge تأیید می‌گردد.

۵. در صورتی که شرایط بالا برقرار نباشد، به این معنی است که نسخه‌ای که از LSA در Database وجود دارد از LSA دریافتی جدیدتر است. در صورتی که کپی موجود در Database دارای LS Age برابر با MaxAge و LS Sequence Number برابر با MaxSequenceNumber باشد، بررسی LSA بدون نیاز به Acknowledge کردن آن پایان می‌یابد. در غیر این صورت، این کپی جدیدتر به همسایه‌ی ارسال کننده از طریق یک بسته‌ی LS Update ارسال می‌گردد.

۴.۴.۱ مقایسه‌ی زمانی دو LSA مشابه

همان‌طور که در فرآیند بالا مشاهده شد، هنگامی که یک مسیریاب به دو مورد از یک LSA برخورد می‌کند باید بتواند تشخیص دهد که کدام جدیدتر است. هر LSA با سه مورد LS Type، Link State ID و Advertising Router مشخص می‌شود. برای مقایسه‌ی جدیدتر بودن دو LSA که این سه مورد آن‌ها یکسان است، از فیلدهای LS Sequence Number، LS Age و LS Checksum استفاده می‌شود. فرآیند مقایسه به صورت زیر است:

- LSAی که دارای مقدار Sequence Number جدیدتری باشد، جدیدتر محسوب می‌شود (LS Sequence Number یک عدد ۳۲ بیتی علامت‌دار است که هر بار یک واحد زیاد می‌شود. می‌توانید فرض کنید که هیچ‌گاه به مقدار ماکزیمم نخواهد رسید). در صورتی که این مقدار برای هر دو یکسان باشد:
- در صورتی که LS Checksumها متفاوت بود، LSAی که دارای Checksum بیشتری باشد، جدیدتر محسوب می‌گردد.
- در غیر این صورت، اگر یکی از LSAها دارای LS Age برابر MaxAge باشد، جدیدتر خواهد بود.
- در غیر این صورت، اگر اختلاف LS Age این دو از مقدار MaxAgeDiff بیشتر باشد، LSAی که LS Age کمتری دارد به عنوان جدیدتر انتخاب خواهد شد.
- در صورتی که هیچ‌کدام از شرایط بالا برقرار نبود، دو LSA یکسان تلقی خواهند شد.

۴.۴.۲ قدم بعدی در فرآیند Flooding

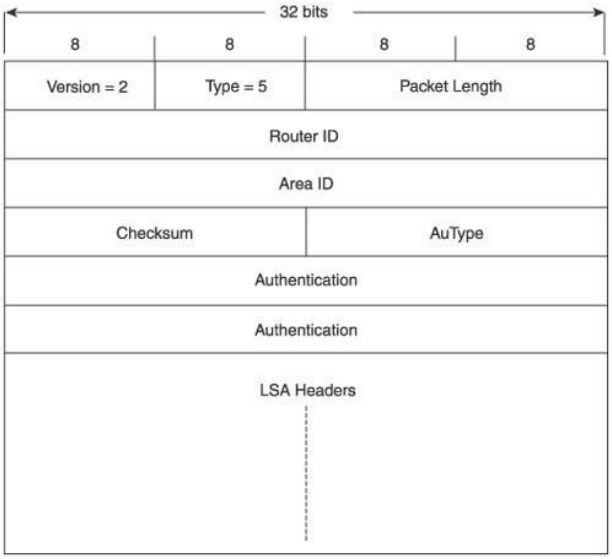
هنگامی که یک LSA جدید دریافت می‌شود، باید در برخی از اینترفیس‌های مسیریاب flood گردد. این بخش مربوط به انتخاب اینترفیس‌ها برای این مورد می‌باشد. همچنین مواردی که در این قسمت به آن‌ها اشاره خواهد شد، برای flood کردن LSAهایی که منشأ آن‌ها خود مسیریاب است هم صادق می‌باشد.

به ازای هر کدام از اینترفیس‌های مسیریاب، برای Flood کردن یک LSA دریافتی، فرآیند زیر صورت می‌پذیرد:

۱. به ازای هر کدام از همسایه‌های متصل به این اینترفیس، فرآیند زیر را خواهیم داشت:
   a. در صورتی که همسایه در پیش از Exchange است، هنوز در Flooding مشارکت نکرده است و باید همسایه‌ی بعدی مورد بررسی قرار گیرد.
   b. درصورتی که هنوز مجاورت کامل بین مسیریاب و همسایه شکل نگرفته است (یعنی همسایه در وضعیت Exchange یا Loading است)، باید لیست LSAهای موردنیاز برای این مجاورت بررسی شود. در صورتی که یک کپی از این LSA در این لیست بود، به این معنی است که این همسایه یک کپی از این LSA را دارد. حال باید این LSA با کپی همسایه مقایسه شود:
      i. در صورتی که LSA جدید، قدیمی‌تر باشد، همسایه‌ی بعدی را بررسی می‌کنیم.
      ii. در صورتی که هر دو کپی یکسان باشد یا LSA جدید، تازه‌تر باشد، آن را از لیست LSAهای موردنیاز حذف می‌کنیم.
   c. در صورتی که LSA از همین همسایه دریافت شده است، همسایه‌ی بعدی را بررسی می‌کنیم.
   d. در این مرحله، مطمئن نیستیم که همسایه یک کپی به روز از این LSA را داشته باشد. بنابراین این LSA را به لیست Retransmission اضافه می‌نماییم تا به صورت قابل اطمینان در بازه‌های زمانی مشخص برای همسایه ارسال شود.
۲. حال اگر در فرآیند قبلی این LSA جدید به هیچ لیست Retransmissionی اضافه نشده بود، در نتیجه نیازی به Flood کردن آن روی این اینترفیس نیست و اینترفیس بعدی باید امتحان شود.
۳. در صورتی که این LSA جدید از همین اینترفیس دریافت شده باشد و مبدأ دریافت DR یا BDR باشد، به احتمال بالایی بقیه مسیریاب‌ها نیز آن را دریافت کرده‌اند؛ بنابراین اینترفیس بعدی بررسی می‌شود.
۴. در صورتی که این LSA جدید از همین اینترفیس دریافت شده باشد و این اینترفیس به عنوان BDR انتخاب شده باشد، اینترفیس بعدی بررسی می‌شود.
۵. در صورتی که به این مرحله رسیدیم، LSA باید روی این اینترفیس Flood شود و بسته‌ی LS Update مربوط باید تولید گردد. مقدار LS Age باید به اندازه‌ی InfTransDelay در هنگام کپی شدن در بسته‌ی LS Update افزایش یابد. آدرس‌های مقصد بسته‌های LS Update در جدول ۱ آمده است.

۴.۴.۳ ارسال و دریافت بسته‌های Acknowledgement

در این تمرین، برای سادگی، بسته‌ی LS Acknowledge در هرجا که نیاز باشد، همان موقع (حالت Direct) ارسال خواهند شد. زمان‌هایی که باید این بسته‌ها ارسال شوند، در توضیح فرآیند Flooding آمده است. آدرس IP مقصد این بسته‌ها را می‌توانید در جدول ۱ بیابید.



تصویر شماره ۹ ـ بسته‌های LS Acknowledge

در صورتی که گیرنده یک Acknowledgement برای یک LSA موجود در Database خودش دریافت کرد، باید آن را از لیست Retransmission حذف نماید. (دقت کنید که به طور عادی، یک LSA از ابتدا در لیست Retransmission قرار می‌گیرد؛ تا زمانی که Acknowledgement برای آن بیاید. این لیست Retransmission برای هر همسایه جداگانه وجود دارد).

۴.۴.۴ ارسال مجدد LSAها

LSAهایی که قرار است به یک عنصر مجاور ارسال شوند، ابتدا در یک لیست به نام لیست Retransmission قرار می‌گیرند. برای اطمینان از رسیدن تمام LSAها، این LSA تا زمانی که Acknowledge نشوند، باید در فواصل زمانی که با RxmtInterval مشخص می‌شوند باز ارسال شوند. دقت کنید که این لیست Retransmission برای هر همسایه است.

ارسال اولین بسته‌های LS Update، به مقصدی که در جدول ۱ ذکر شده است انجام می‌شود؛ اما بسته‌هایی که قرار است باز ارسال شوند، باید دقیقاً به آدرس Unicast خود همسایه ارسال شوند. دقت کنید که اگر مسیریاب همسایه به دلیلی از شبکه خارج شود، Retransmission تا زمانی که مسیریاب توسط پروتکل HELLO (تمرین قبل) از لیست همسایه‌ها خارج نشده است، ادامه می‌یابد. در زمانی که این همسایگی از بین برود، باید این لیست Retransmission نیز از بین برود.

۴.۴.۵ تولید LSAها

تا اینجا با نحوه‌ی Flooding برای LSAها آشنا شدیم. ولی در این مرحله می‌خواهیم ببینیم که LSAها در چه زمان‌هایی تولید می‌شوند؟

شرایط زیر باعث می‌شوند تا یک مسیریاب اقدام به ایجاد LSA کند:

۱. مقدار LS Age برای یکی از LSAهای تولید شده توسط خود مسیریاب به میزان LSRefreshTime برسد. در این صورت یک نسخه‌ی جدید از این LSA تولید می‌شود (هرچند که ممکن است هیچ تفاوتی بین LSA قبلی و جدید نباشد). به این ترتیب تضمین می‌شود تا تمام LSAها به صورت تناوبی تولید و Flood شوند.

بر خلاف مورد اول، شرایط زیر تنها در صورتی اقدام به ایجاد LSAهای جدید و Flood آن‌ها می‌کنند که LSA جدید متفاوت از نسخه‌ی قبلی خود باشند. دقت کنید که بین تولید دو نسخه متفاوت از یک LSA باید حداقل MinLSInterval فاصله باشد.

۲. وضعیت هرکدام از اینترفیس‌های مسیریاب تغییر کند.
۳. DR شبکه تغییر نماید؛ در صورتی که مسیریاب خودش DR جدید باشد، باید Network LSA جدید نیز تولید شود.
۴. یکی از مسیریاب‌های همسایه به وضعیت FULL برود. در صورتی که مسیریاب DR باشد، باید علاوه بر Router LSA، Network LSA هم تولید کند.

۵ ایجاد جدول مسیریابی

با اجرای مراحلی که در پیش ذکر شد، همه‌ی مسیریاب‌ها بعد از طی یک بازه‌ی زمانی، دارای Link State Database یکسان خواهند شد. این LS Database شامل تعدادی Router LSA و Network LSA است. Network LSAها شبکه‌های موجود در ناحیه و روترهای موجود در آن را مشخص می‌کند. به این ترتیب، تمام شبکه‌های موجود در ناحیه و مسیریاب‌های آن‌ها توسط Network LSAهایی که DRهای هر شبکه تولید کرده‌اند، مشخص می‌شود.

از طرف دیگر، هر Router LSA وضعیت Linkهای یک مسیریاب را مشخص می‌کند. به این ترتیب، با داشتن مجموعه‌ی کامل این دو نوع LSA می‌توان توپولوژی شبکه را به طور کامل در اختیا داشت. از آنجایی که LS Database برای تمام مسیریاب‌ها یکسان است، بنابراین همه‌ی مسیریاب‌ها توپولوژی یکسانی را از شبکه می‌بینند.

حال هر مسیریاب، با اجرای یک الگوریتم کوتاه‌ترین مسیر؛ مثلاً Dijkstra با شروع از خودش به عنوان ریشه، می‌تواند یک درخت Shortest Path ایجاد کند. به کمک این درخت (که هزینه‌ی هر یال در واقع هزینه‌ی اینترفیس مربوط است و در Router LSA آمده است)، مسیرهای بهینه تا تمام بقیه‌ی مسیریاب‌ها محاسبه می‌شود. به این ترتیب، به سادگی جداول مسیردهی قابل تشکیل است.

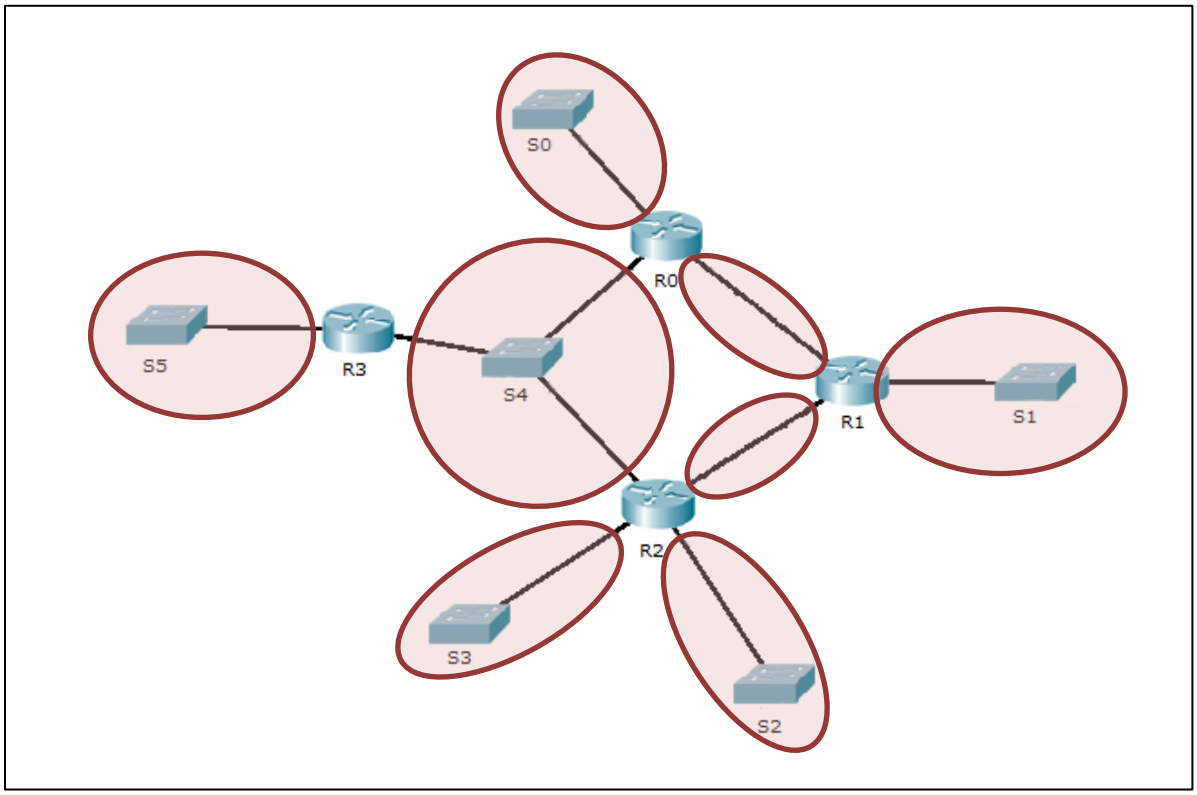
برای اطلاعات بیشتر در این باره و نحوه‌ی دقیق تولید این درخت می‌توانید به لینک زیر مراجعه کنید:

http://tools.ietf.org/html/rfc2328#page-160

در این تمرین ساخت جداول مسیریابی و مسیریابی بسته‌ها مد نظر نیست و صرفاً هدف پیاده‌سازی همگام‌سازی LS Databaseهاست.

۶ برنامه‌ی مسیریاب

هدف نهایی این تمرین، یکسان سازی LS Database بین تمامی مسیریاب‌ها و اجرای درست عملیات Flooding است. توپولوژی شبکه‌ی نمونه به صورت زیر می‌باشد.



تصویر شماره ۱۰ ــ توپولوژی شبکه‌ی نمونه

شبکه‌های مستقیماً متصل

هر مسیریاب تعدادی interface دارد. برای پیدا کردن این تعداد می‌توانید از متغیر countOfInterfaces که در کلاس Machine تعریف شده است استفاده کنید.

پروتکل مسیریابی OSPF برای شروع به کار روی هر مسیریاب، به داشتن آدرس و Mask شبکه‌هایی که مستقیماً به آن مسیریاب متصل هستند نیاز دارد. برای این منظور، باید با استفاده از IP Address و Subnet Mask هر یک از Interfaceها، Network Address و Network Mask شبکه‌های متصل به مسیریاب را محاسبه و نگهداری کنید. در کلاس Interface می‌توانید متغیرهای مورد نیاز برای این کار را پیدا کنید.

مثلاً اگر برای iface[0] داشته باشیم:

IP/Mask = 213.233.168.17/29

می‌توانیم نتیجه بگیریم که یکی از شبکه‌های متصل به مسیریاب، شبکه‌ی زیر است:

IP/Mask = 213.233.168.16/29

مقادیر اولیه

هر مسیریاب یک ID یکتا دارد که یک عدد ۳۲ بیتی است و با آن خود را به سایر مسیریاب‌ها معرفی می‌کند. برای این کار، از iface[0].getIp() استفاده نمایید. باید دقت داشته باشید که این عدد برای هر مسیریاب ثابت باشد و برای بسته‌های مختلفی که روی interfaceهای مختلف ارسال می‌کند، متفاوت نباشد.

بقیه‌ی مقادیر اولیه مورد نیاز، به شکل زیر از طریق getCustomInformation() قابل دسترسی هستند.

```
<area-id>
<hello-interval>
<router-dead-interval>
<retransmit-interval>
<iface[0]-cost>
<iface[0]-priority>
<iface[1]-cost>
<iface[1]-priority>
…
<iface[n]-cost>
<iface[n]-priority>
```

محیط

در این تمرین نیز، مانند تمرین قبل، از محیط پرتو برای اجرای برنامه شما استفاده خواهد شد. برای اطلاعات بیشتر به مستند «راهنمای چارچوب کاربر» مراجعه فرمایید.

نکات ضروری

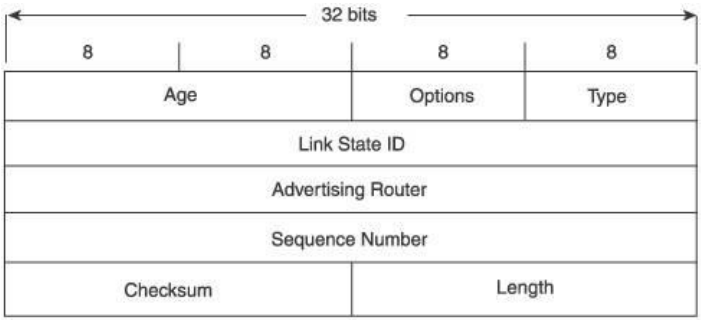
- در صورتیکه هر مشکل یا پرسشی داشتید که فکر می‌کنید پاسخ آن برای همه مفید خواهد بود، لطفاً آن را به گروه پستی درس ارسال کنید.
- از فرستادن جواب تمرین به گروه پستی جداً خودداری کنید.
- فرستادن کل یا قسمتی از برنامه‌تان برای افراد دیگر، یا استفاده از کل یا قسمتی از برنامه‌ی فردی دیگر به نام خود، تقلب محسوب می‌شود.
- پس از اتمام کارتان لازم است که پوشه user را به همراه Makefile فشرده کرده و بر روی سیستم خودکار داوری[5] upload کنید.

[5] http://partov.sharif.edu/

۷ ضمیمه‌ی ۱: بسته‌های LSA

۷.۱ سرآیند مشترک بسته‌های LSA

بسته‌های LSA یازده نوع مختلف دارند که فقط از دو نوع آنها (Router-LSA و Network-LSA) در این تمرین استفاده خواهد شد. این بسته‌ها سرآیندی ۲۰ بایتی دارند و با توجه به نوعشان، محتوای بسته تعیین می‌شود. سرآیند این بسته‌ها به شکل زیر است:



تصویر شماره ۱۱

بخش‌های مختلف این سرآیند به صورت زیر است:

بخش	توضیحات
Age	زمان به ثانیه از وقتی که این LSA تولید شده است.
Options	برای قابلیت‌های اختیاری پروتکل OSPF که در این تمرین مد نظر نمی‌باشد.
Type	مشخص کننده‌ی نوع LSA که در این تمرین یکی از دو نوع Router LSA (مقدار ۱) و Network LSA (مقدار ۲) می‌باشد.
Link State ID	مشخص کننده‌ی محدوده‌ای است که توسط این LSA توصیف می‌گردد. محتویات این بخش بستگی به نوع LSA دارد. • برای Type 1 (Router LSA): برابر با مقدار Router IDی که این LSA را تولید کرده است. • برای Type 2 (Network LSA): برابر با آدرس IP اینترفیس Designated Router شبکه.
Advertising Router	مقدار Router ID مسیریابی که این LSA را تولید کرده است. مثلاً برای Network LSAها، این مقدار برابر با Router ID مسیریاب برگزیده در شبکه می‌باشد.
LS Sequence Number	برای تشخیص LSAهای قدیمی و تکراری مورد استفاده قرار می‌گیرد. جزئیات بیشتر در مورد این بخش در آینده شرح داده خواهد شد.

LS checksum	Checksum برای تمام محتویات LSA به همراه سرآیند آن، به جز فیلد Age. جزئیات بیشتر در مورد این بخش در آینده شرح داده خواهد شد.
Length	طول LSA به بایت که شامل ۲۰ بایت سرآیند نیز می‌گردد.

هنگامی که یک LSA جدید تولید می‌شود، مقدار LS Sequence Number آن زیاد می‌شود، مقدار LS Age برابر با صفر قرار داده می‌شود، مقدار LS Checksum برای آن محاسبه می‌شود و به پایگاه‌داده‌ی Link State افزوده می‌گردد و سپس در اینترفیس‌های مربوط Flood می‌گردد.

۷.۱.۱ فیلد LS Age

این فیلد نشان دهنده‌ی Age برای LSA به ثانیه است و باید به صورت یک عدد ۱۶ بیتی فاقد علامت تعبیر شود. هنگامی که LSA تولید می‌شود، مقدار این فیلد برابر با صفر است. در عملیات Flooding با عبور بسته از هر Hop، مقدار آن باید به اندازه‌ی InfTransDelay افزایش یابد. همچنین مقدار این فیلد در زمانی که LSA در LS Database مسیریاب قرار دارد نیز باید زیاد شود. هر ثانیه مقدار این فیلد یک واحد افزایش می‌یابد. مقدار MaxAge برابر با «یک ساعت» در نظر گرفته می‌شود. InfTransDelay را نیز برابر با ۱۰ ثانیه در نظر بگیرید.

برای سادگی می‌توانید فرض کنید که برنامه‌های شما در زمانی کمتر از MaxAge اجرا می‌شوند و LSAها به MaxAge نخواهند رسید.

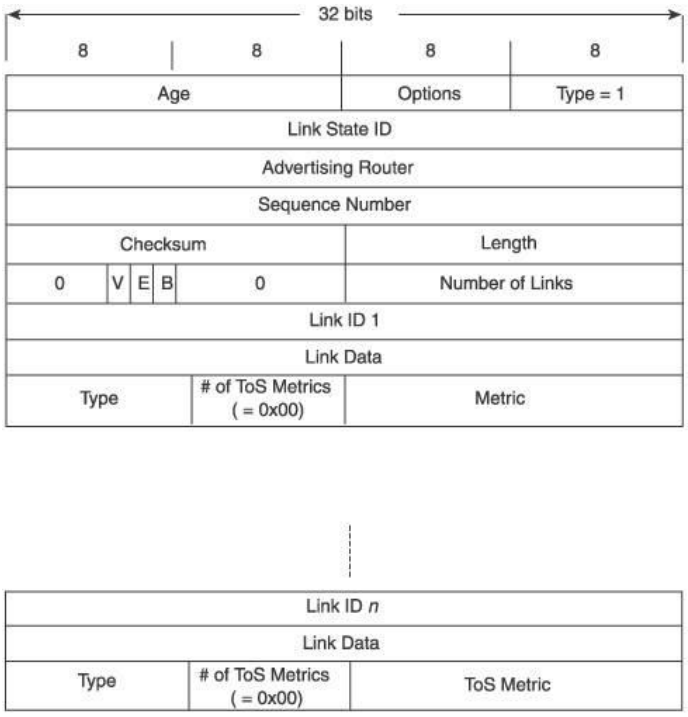
۷.۱.۲ فیلد LS Checksum

برای آشنایی با نحوه‌ی محاسبه‌ی این فیلد به لینک زیر مراجعه کنید:

http://tools.ietf.org/html/rfc2328#page-121

۷.۲ Router-LSA

هر مسیریاب برای کل ناحیه‌ای که در آن قرار دارد یک Router LSA تولید می‌نماید.



تصویر شماره ۱۲ – Router LSA

بخش‌های مختلف این LSA به صورت زیر است:

بخش	توضیحات
V, E, B	تمامی این بیت‌ها در این تمرین صفر هستند.
Number of Links	تعداد لینک‌های مسیریابی که در این LSA شرح داده شده‌اند.
به ازای هر کدام از لینک‌ها، فیلدهای زیر توصیف کننده‌ی آن لینک می‌باشند.	
Type	اگر اینترفیس در وضعیت Waiting باشد، مقدار این فیلد برابر با 3 (وضعیت Stub Network) خواهد بود. در غیر این صورت؛ یک DR برای شبکه انتخاب شده است. در صورتی که این مسیریاب با DR «به طور کامل» مجاور شده باشد (Fully Adjacent)، یا خودش DR باشد و حداقل با یک مسیریاب دیگر به طور کامل مجاور شده باشد، مقدار این فیلد برابر با 2 (وضعیت Transit Network) خواهد بود. در صورتی که هیچ‌کدام از این شرط‌ها برقرار نبود، Type بر اساس فرض Waiting بودن (وضعیت بالا) تعیین می‌شود.
Link ID	در صورتی که Type برابر با 3 باشد، مقدار این فیلد برابر با آدرس IP شبکه است. در صورتی که Type برابر با 2 باشد، این مقدار برابر با آدرس IP اینترفیس مربوط DR خواهد بود.

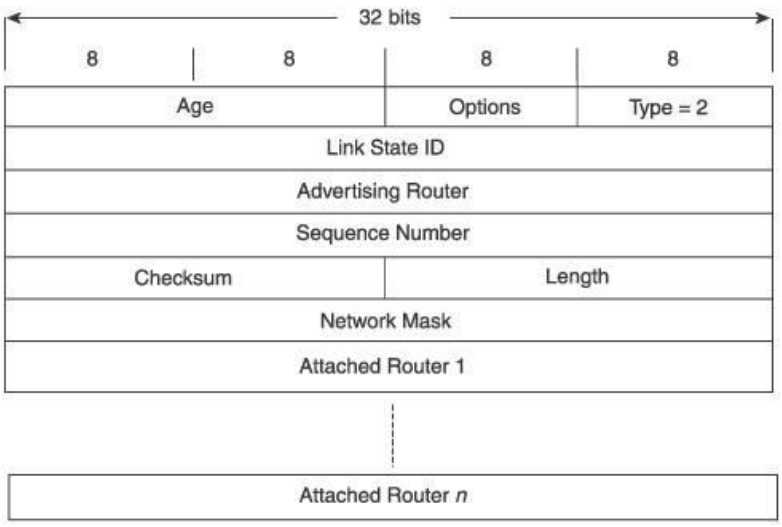
Link Data	در صورتی که Type برابر با 3 باشد، مقدار این فیلد برابر با Mask شبکه است. در صورتی که Type برابر با 2 باشد، مقدار این فیلد برابر با آدرس IP اینترفیس خود مسیریاب خواهد بود.
# TOS Metrics	برابر با تعداد متریک‌هایی است که غیر از متریک اصلی برای این لینک بیان می‌شوند. در این تمرین تنها همان متریک اصلی را ـ که هزینه‌ی Link است ـ خواهیم داشت. بنابراین این مقدار برابر با صفر است.
Metric	هزینه‌ی استفاده از این لینک را مشخص می‌کند و برابر با هزینه‌ی تعیین شده برای اینترفیس است.

۷.۳ Network LSA

نوع دیگر LSAهایی که در این تمرین مورد استفاده قرار می‌گیرد، Network LSAها هستند. یک Network LSA برای هر شبکه‌ی broadcastی که دو یا بیشتر مسیریاب داشته باشد، تولید می‌شود. وظیفه‌ی تولید این LSA بر عهده‌ی Designated Router می‌باشد. بنابراین DR زمانی اقدام به تولید این LSA می‌کند که حداقل با یک مسیر یاب دیگر به طور کامل مجاور شده باشد. این LSA وضعیت تمام Routerهای متصل به شبکه (شامل خود Designated Router) را توصیف می‌کند. دقت کنید که فاصله‌ی شبکه از تمام مسیریاب‌های خودش «صفر» است. بنابراین این LSA فاقد فیلد Metric است.

این LSA در کل ناحیه‌ای که شامل این شبکه است، پخش می‌شود (ر. ک. به Flooding).

همان‌طور که در قبل گفته شد، مقدار Link State ID برای Network LSA، برابر با IP اینترفیس مربوط Designated Router است؛ با استفاده از این مقدار و مقدار Mask که در این بسته آمده است، IP شبکه به سادگی بدست می‌آید.



تصویر شماره ۱۳ ـ Network LSA

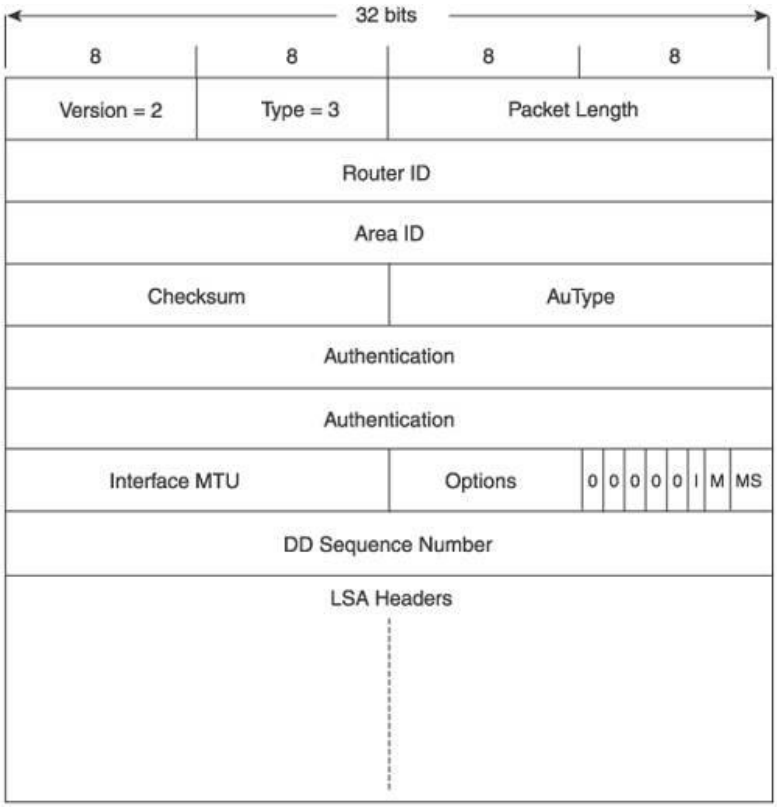
بخش	توضیحات
Network Mask	Mask مربوط به آدرس‌های IP این شبکه. مثلاً برای یک شبکه‌ی کلاس A، این مقدار برابر است با 0xff000000.
Attached Router *i*	Router ID برای هر کدام از مسیریاب‌های موجود در شبکه. دقت کنید که تنها مسیریاب‌هایی در این لیست قرار می‌گیرند که به طور کامل با DR مجاور شده باشند. خود DR نیز در این لیست حضور دارد. تعداد این مسیریاب‌ها می‌تواند به سادگی از روی فیلد Length محاسبه شود.

۸ ضمیمه‌ی ۲: بسته‌های مورد استفاده در فرآیند Flooding

۸.۱ بسته‌ی Database Description

این بسته‌ها، بسته‌های نوع ۲ پروتکل OSPF می‌باشند. این بسته‌ها زمانی ارسال می‌شوند که قرار است یک مجاورت بین دو همسایه شکل بگیرد. وظیفه‌ی این بسته‌ها توصیف محتویات Link-state Database مسیریاب می‌باشد. ممکن است برای توصیف Database نیاز به چند عدد از این بسته‌ها باشد.

فرمت کلی این بسته به شکل زیر می‌باشد:



تصویر شماره ۱۴ ـ بسته‌ی DD

۳۲ بایت اول سرآیند مشترک تمام بسته‌های OSPF است. سایر قسمت‌های مرتبط با بسته‌های DD به صورت زیر می‌باشد:

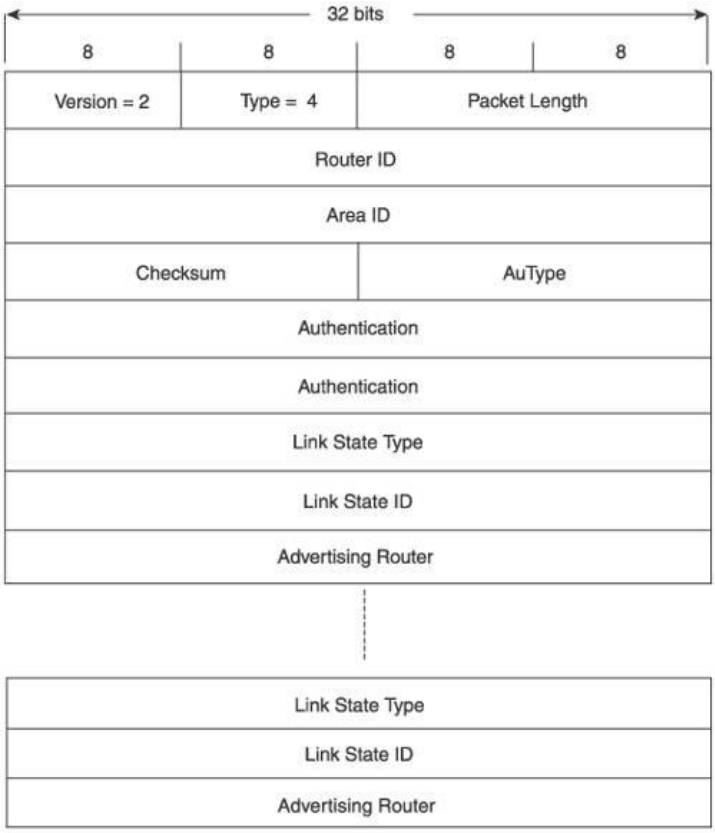
بخش	توضیحات
Interface MTU	حداکثر سایز بسته‌های IP بدون اینکه نیاز به fragmentation داشته باشند. برای پروتکل لایه‌ی لینک Ethernet این مقدار برابر 1500 است.
Options	برای قابلیت‌های اختیاری پروتکل OSPF که در این تمرین مد نظر نمی‌باشد.
I-bit	بیت Init. وقتی که این بیت ۱ باشد، یعنی این اولین بسته‌ی DD می باشد.
M-bit	بیت More. وقتی که این بیت ۱ باشد، به این معنی است که بسته‌های DD به پایان نرسیده‌اند.
MS-bit	بیت Master/Slave. اگر این بیت ۱ باشد، یعنی مسیریاب در طی فرآیند Database Exchange، Master است. در غیر این صورت، مسیریاب Slave می‌باشد.
DD Sequence Number	برای دادن ترتیب به بسته‌های Database Description مورد استفاده قرار می‌گیرد. مقدار اولیه این فیلد (که در بسته‌ی DD با بیت I برابر با ۱ تعیین می‌شود) باید یکتا باشد؛ مثلاً به صورت تصادفی تولید شود. با ارسال هر بسته‌ی DD مقدار این فیلد یکی افزایش می‌یابد.
LSA Headers	در ادامه‌ی بسته سرآیند LSAهایی که مسیریاب دارای آن‌هاست خواهد آمد. می‌توانید فرض کنید که

توپولوژی به‌گونه‌ای است که تمام سرآیندها در یک بسته جا می‌شوند.

۸.۲ بسته‌های Link-State Request

این بسته‌ها، بسته‌های نوع ۳ پروتکل OSPF می‌باشند. بعد از تبادل کردن DDها، یک مسیریاب ممکن است دریابد که بخشی از Link-State Database آن به روز نیست. این مسیریاب به کمک این بسته‌ها از مسیریاب دیگر درخواست می‌کند تا LSAهای جدیدتر را دریافت کند.

فرمت این بسته‌ها به شکل زیر است:



تصویر شماره ۱۵ ـ بسته‌ی LS Request

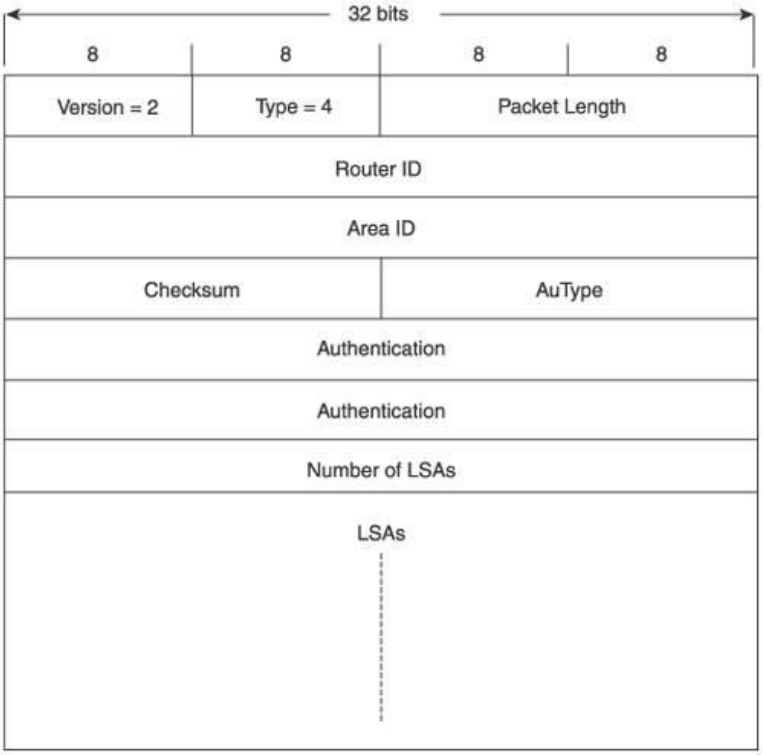
به ازای هر یک از LSAهای درخواستی، سه فیلد زیر باید مشخص شوند. این سه فیلد متناظر با همان فیلدهایی است که در سرآیند LSA آمده است. تعداد این درخواست‌ها به سادگی از روی طول بسته قابل محاسبه است.

بخش	توضیحات
LS Type	دقیقاً متناظر با همین فیلدها در سرآیند LSA
Link State ID	
Advertising	

Router

۸.۳ بسته‌های Link-State Update

این بسته‌ها، بسته‌های نوع ۴ پروتکل OSPF به حساب می‌آیند. فرمت کلی این بسته‌ها به شکل زیر است:

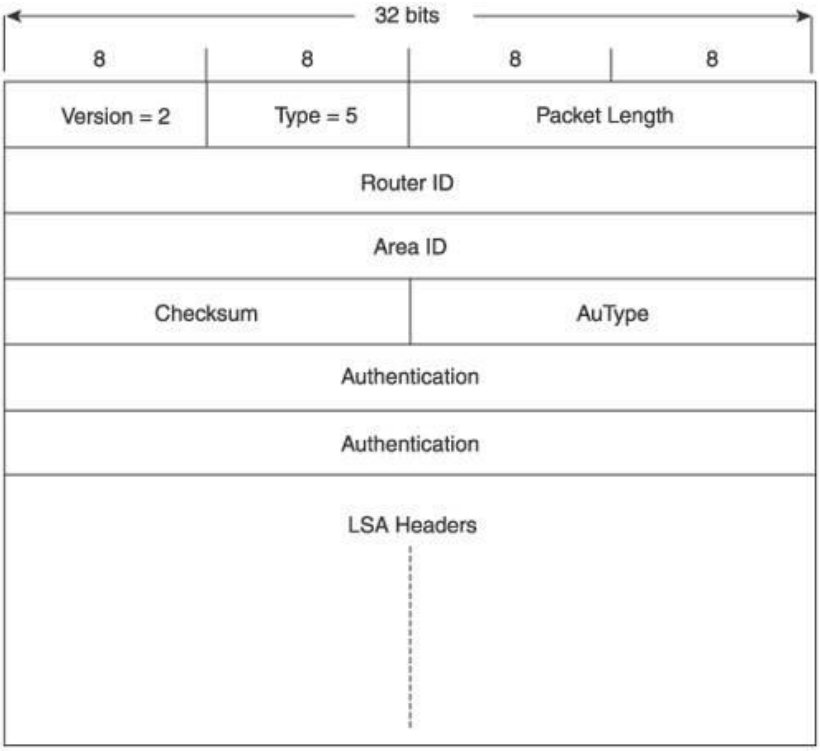


تصویر شماره ۱۶ ـ بسته‌ی LS Update

بخش	توضیحات
Number of LSAs	تعداد LSAهایی است که در این بسته آمده است.
LSAs	LSAها به طور کامل به همراه سرآیند در ادامه‌ی بسته قرار می‌گیرند.

۸.۴ بسته‌های Link-State Acknowledgement

این بسته‌ها، بسته‌های نوع ۵ پروتکل OSPF هستند و برای اطمینان از رسیدن بسته‌های LSA مورد استفاده قرار می‌گیرند. فرمت این بسته‌ها به صورت زیر می‌باشد:



تصویر شماره ۱۷ ـ بسته‌ی LS Acknowledge

محتویات بسته مجموعه‌ای از سرآیندهای LSA است.