«« مهندسی اینترنت »»

مؤسسه آموزش عالی ادیبان گرمسار

جزوه استاد ایران پور

تهیه و تنظیم :

سرکار خانم رزمی

۱۳۹۰

فهرست مطالب

عنوان **صفحه**

بخش اول:

مبانی شبکه های کامپیوتری

تعریف شبکه کامپیوتری: مجموعه ای از کامپیوترهای مستقل است،که از طریق یک رسانه انتقال با یکدیگر به تبادل اطلاعات و داده می پردازند،یک زیر ساخت برای حمل و جابجایی اطلاعات به شمار می آید که از سه دهه ی قبل وجود داشته است.

تعریف وب: معماری سازمان دهی و دسترسی به اطلاعات توزیع شده در سطح جهان(یا به عبارتی یک ابزار بر روی اینترنت است)که از عمر آن حدود ۱۵ سال می گذرد.

تعریف اینترنت : مجموعه ای از شبکه های مستقل و مرتبط با یکدیگر است که ارتباطات همگانی را میسر کرده است.

اینترانت: شبکه ای داخلی است که از پروتکل های مرتبط با اینترنت ، مخصوصاً تکنولوژی وب برای سازمان دهی شبکه استفاده می کند. هیچ لزومی ندارد که اینترانت حتماً به اینترنت متصل باشد. مثلاً اداره پست می تواند برای خود شبکه ای داخلی و مستقل از شبکه اینترنت تدارک ببیند ، ولی در پیاده سازی این شبکه از پروتکل های حاکم بر اینترنت (TCP/IP/Http/WWW)و پایگاه داده های مبتنی بر وب استفاده کند.

کاربردهای شبکه کامپیوتری:

۱- **اشتراک منابع:** به معنای فراهم آوردن و به اشترک گذاشتن سخت افزار ، نرم افزار و داده های مورد نیاز در شبکه است به گونه ای که کاربران بتوانند به راحتی از این منابع استفاده کنند ، به عنوان مثال یک چاپگر در شبکه می تواند در اختیار کل کاربران باشد.

۲- **حذف محدودیت های جغرافیایی در تبادل داده ها:** دهکده ی جهانی اینترنت ، فواصل جغرافیایی را بی معنا کرده است با استفاده از شبکه های کامپیوتری شما می توانید در کسری از ثانیه به منابع اطلاعاتی موجود در فواصل هزاران کیلومتری خود دسترسی داشته باشید.

۳- **کاهش هزینه ها:** به کارگیری شبکه های کامپیوتری نه تنها در وقت صرفه جویی می کند ، بلکه هزینه ها را نیز در تمام جوانب کاهش می دهد.به عنوان مثال یک چاپگر یا نرم افزار برای ۲۰ نفر به جای ۲۰ چاپگر و نرم افزار برای ۲۰ نفر.

۴- **بالا رفتن قابلیت اعتماد سیستم ها:** شبکه های کامپیوتری به گونه ای طراحی می شوند که وقتی یکی از آنها مختل شود بقیه ی شبکه از هم نمی پاشد،ذخیره سازی بانکهای اطلاعاتی موجود در شبکه یک سازمان بر روی چند ماشین مستقل به عنوان سیستم های پشتیبان این فایده را دارد که در صورت خرابی یکی از آنها دیگری جایگزین آن می شود.

۵- **افزایش کارایی سیستم:** بهره گیری از شبکه می تواند کارایی سیستم را از لحاظ سرعت دسترسی به اطلاعات و سرعت پردازش و ذخیره و بازیابی اطلاعات افزایش دهد.

دسته بندی شبکه ها از دیدگاه تکنولوژی انتقال:

ازدیدگاه تکنولوژی انتقال، دو دسته کلی از شبکه قابل تعریف است:

۱- **شبکه های پخش فراگیر(Broad Cast)** : در این شبکه ها انتقال اطلاعات از طریق یک کانال فیزیکی که بین تمام ایستگاه های شبکه مشترک است انجام می شود.

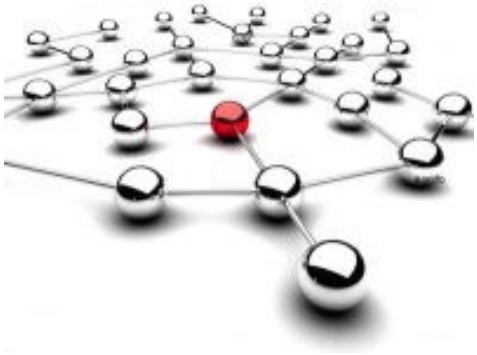
همه ایستگاه ها موظفند به طوردائم به خط گوش بدهند و برای ارسال نیز مجبورند اطلاعات را بر روی همین کانال منتقل نمایند.

بنابراین در چنین شبکه هایی هر ایستگاه باید یک آدرس یکتا داشته باشد ، تا گیرنده پیام بتواند از بین پیام هایی که روی شبکه مبادله می شود پیام مربوط به خودش را تشخیص دهد، در این نوع شبکه امکان ارسال پیام های فراگیر وجود دارد. پیام های فراگیر، پیام هایی هستند که از یک ایستگاه برای تمام یا یک گروه خاص از ایستگاه ها ارسال شود.

استفاده از کانال های مشترک برای انتقال اطلاعات بین ایستگاه های شبکه دارای مشکلاتی مانند مدیریت پیچیده کانال، امنیت کم و کارایی پایین است ، اما بسیار مقرون به صرفه بوده و به صورت گسترده از آن استفاده می شود ، مانند شبکه های ماهواره ای.

۲- **شبکه های نقطه به نقطه یا Point-to-point :**

در این شبکه ها، بین دو ماشین در شبکه یک کانال فیزیکی و مستقیم وجود دارد وهیچ ماشین دیگری به آن کانال متصل نخواهد بود.



منظور از شبکه های نقطه به نقطه آن نیست که بین هر دو ماشین از شبکه حتما یک خط مستقیم وجود دارد، بلکه به این معناست که اگر چنین کانالی وجود داشته باشد،آن کانال اختصاصی و مستقیم است، بر خلاف شبکه های با کانال مشترک که مسیر ارتباطی بین ایستگاه ها یکتا است ، در این شبکه ها مسیرهای گوناگونی بین دو ماشین برقرار خواهد بود،لذا بحث انتخاب بهترین مسیر از بین مسیرهای مختلف هم مطرح خواهد شد که تحت عنوان مسیریابی به آن خواهیم پرداخت.

دسته بندی شبکه ها از دیدگاه مقیاس بزرگی:

دیدگاه دوم در دسته بندی و تفکیک شبکه ها مقیاس شبکه و وسعت ناحیه تحت پوشش آنها است ، شبکه های کامپیوتری از دیدگاه مقیاس بزرگی به ۵ دسته ی PAN, LAN,MAN,RAN,WAN تقسیم می شوند.

شبکه های شخصی یا PAN (Personal Area Network):

این شبکه ها درمحدوده ای کمتر از ۱۰ متر شکل می گیرند و مالکیت فردی دارند این رده از شبکه برای اتصال دستگاه های شخصی وخانگی مثل کامپیوتر،تلفن همراه،فکس،چاپگر،دوربین وغیره به یکدیگر کاربرد دارد.تکنولوژی USB(سیمی)وبلوتوث (بی سیم) برای این رده از شبکه ها توسعه داده شده است.

شبکه های محلی LAN (Local Area Network):

این شبکه ها در فواصل جغرافیایی محدود(معمولاً تا ۲ کیلومتر) و تحت تملک سازمان های کوچک ، ادارات ، نهادها و محیط های آموزشی نصب و راه اندازی می شوند.

شبکه های بین شهری MAN (Metropolitan Area Network):

این شبکه ها در یک منطقه وسیع مثل یک شهر پیاده سازی می شوند و از لحاظ تکنولوژی بیشتر به LAN شبیه هستند.

شبکه های منطقه ای RAN(Regional Area Network):

این شبکه ها در یک کشور (مثل شبکه های استانی یا ایالتی) وعموماً با هدف ارائه خدمات خاص پیاده سازی می شوند.

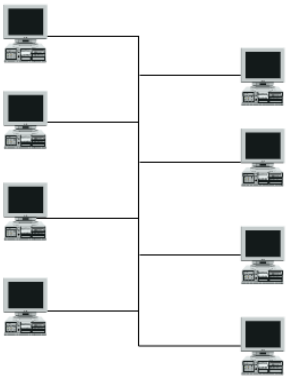
شبکه گسترده WAN(Wide Area Network):

این شبکه ها در کشور، قاره ، یا جهان پیاده سازی می شوند و شبکه های محلی و بین شهری را به هم وصل می نمایند.

تعریف توپولوژی شبکه:

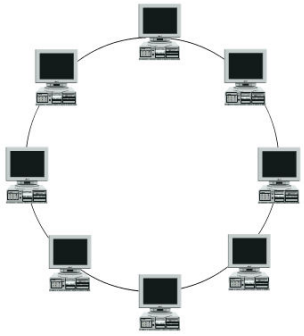
چگونگی به هم بندی و اتصال ماشین ها به کانال انتقال و ایجاد یک شبکه واحد را توپولوژی شبکه می گویند.

توپولوژی های رایج برای شبکه های محلی عبارتند از:



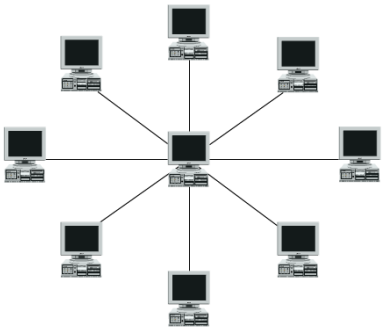
۱) توپولوژی خطی (BUS) : در این نوع توپولوژی تمام ماشین ها از طریق یک کانال فیزیکی مشترک به یکدیگر متصل شده اند و هرگونه تبادل اطلاعات از طریق این کانال انجام خواهد شد.

۲) توپولوژی حلقه (Ring) :



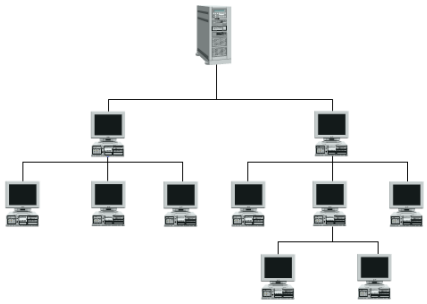
در این توپولوژی، ایستگاه ها در یک ساختار بسته حلقوی به یکدیگر متصل می شوند،جهت جریان اطلاعات یکی از دو حالت ساعت گرد یا پاد ساعت گرد است و برای آن که اطلاعات از یک ایستگاه به ایستگاه غیر مجاور آن در حلقه منتقل شود، باید ماشینهایی که در مسیر هستند بیت های داده را دریافت و درخروجی خود تکرار کنند تا در نهایت اطلاعات به مقصد برسد. ارتباط هر ایستگاه با ایستگاه بعدی خود درحلقه یک طرفه است، و اگر یک ایستگاه بخواهد به ماشین قبلی خود در حلقه بسته ای از داده ها را بفرستد، آن بسته باید یک دورکامل درحلقه گردش کند تا به ایستگاه مورد نظر برسد.

۳) توپولوژی ستاره (Star) :



در این توپولوژی ارتباط تمامی ماشین های شبکه از طریق یک گره مرکزی برقرار می شود. این گره می تواند یک سوئیچ بسیار سریع و هوشمند ، یا یک هاب معمولی و یا حتی یک کامپیوتر باشد.

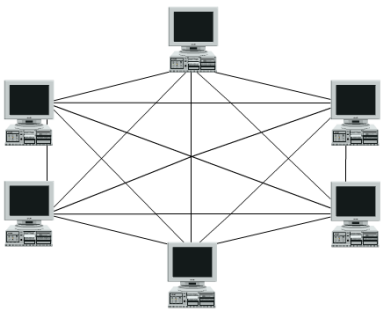
۴) توپولوژی درختی یا سلسله مراتبی :

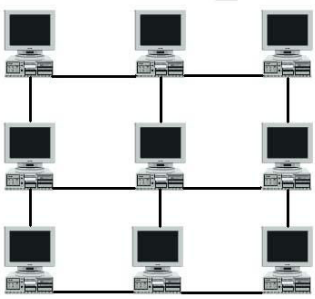


در این نوع از توپولوژی ماشین ها در یک الگوی درختی به یکدیگر متصل می شوند. برگ های این درخت، همان ماشین ها و گره های میانی ، عناصر ارتباطی(مثل سوئیچ یا هاب)هستند.هرگاه دو ماشین همزاد باشند(یعنی از یک عنصر میانی منشعب شده باشند) ارتباط آنها توسط گره پدرشان برقرار می شود و در غیر این صورت برقراری ارتباط آنها در سطوح بالاتر انجام می گیرد.

۵)توپولوژی با اتصال کامل و توپولوژی توری شکل:

در توپولوژی با اتصال کامل بین هر دو ماشین از شبکه ، یک کانال انتقال مستقیم و اختصاصی وجود دارد. در توپولوژی توری شکل هر ماشین حداقل با چهار ماشین همسایه ی خود دارای کانالی اختصاصی است.





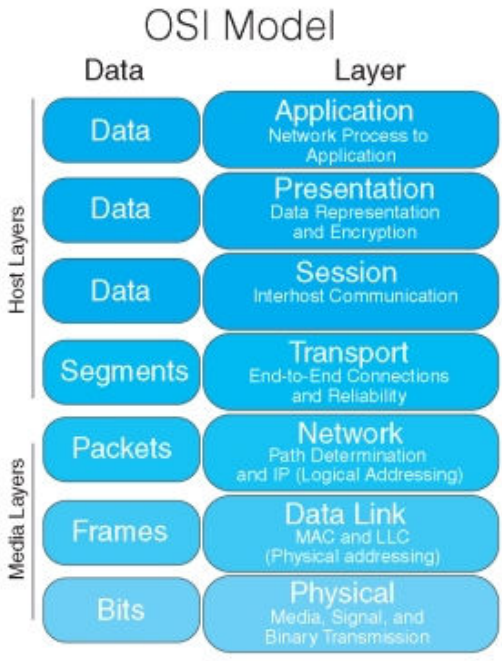
معماری و عملکرد لایه ای شبکه: به دلیل پیچیدگی بسیار زیاد و گستره ی مؤ لفه هایی که پیکره ی یک شبکه کامپیوتری را تشکیل می دهند، معماری یک شبکه کامپیوتری به صورت لایه ای طراحی می شود، وظیفه ی هر لایه ارائه ی سرویس های خاص به لایه ی بالاتر است، این سرویس ها فارغ از نوع پیاده سازی آن ها و باپنهان نگاه داشتن جزئیات کار از لایه های بالا تر ارائه می شود.

تعریف پروتکل : عبارت است ازکلیه ی قرارداد های توافق شده بین دو لایه ی همتا برای برقراری و پیشبرد یک ارتباط. این قراردادها عبارتند از پایبندی به یک الگوی دقیق و واحد در مورد قالب هر پیام ، مفهوم و تعبیر پیام ها ، شکل و زمان بندی صحیح مبادله پیام ها.

تعریف پشته ی پروتکلی: به کلیه ی پروتکل های تعریف شده در لایه های یک شبکه ی کامپیوتری که عملکرد صحیح شبکه را تضمین می کند ،پشته ی پروتکلی گفته می شود.

تعریف PDU(Protocol Data Unit) : در معماری لایه ای شبکه یک قطعه داده که از لایه ی K+1 تحویل لایه ی K ام می شود ، پس از پردازش ها و تراکنش های لازم با لایه ی همتا از ماشین گیرنده ، درقالب یک ساختمان داده استاندارد و مشخص سازماندهی شده ، تحویل لایه ی زیرین می شود. تعریف دقیق و جامع این ساختمان داده در توصیف پروتکل مربوطه آمده است. به قطعه ای که در هر لایه سازمان دهی و تحویل لایه ی زیرین می شود به طورعام PDU گفته می شود.

کپسوله سازی اطلاعات: طبق آنچه که اشاره شد، هرلایه پس از دریافت یک قطعه داده از لایه ی فوقانی، آن را در قالب یک PDU سازماندهی کرده و تحویل لایه زیرین می دهد.تشکیل PDU مستلزم اضافه کردن چندین فیلد اطلاعاتی به ابتدا و گاهی به انتهای قطعه داده است. با افزوده شدن این فیلدها ، قطعه داده دارای هویت شده و لایه ی همتا در ماشین گیرنده قادر به درک و پردازش آن خواهد بود ، به این فرآیند کپسوله سازی اطلاعات گفته می شود.



مدل مرجع OSI : برای آنکه طراحی شبکه ها سلیقه ای و پیچیده نشود ، سازمان جهانی استاندارد سازی **(ISO)** مدل هفت لایه ای برای شبکه ارائه کرد ، به گونه ای که وظایف و خدمات شبکه در هفت لایهء مجزا تعریف و ارائه می شود. این مدل هفت لایه ای ، مدل مرجع **OSI** نام گرفت.

OSI (Open System Interconnection)

ISO (Intenational Standard Organization)

در این استاندارد کل و ظایف و خدمات یک شبکه در هفت لایه تعریف شده است.

1) Physical Layer
2) Data Link Layer
3) Network Layer
4) Transport Layer
5) Session Layer
6) Presentation Layer
7) Application Layer

لایه ی فیزیکی : وظیفه اصلی این لایه انتقال بیت ها بر روی کانال مخابراتی است. در این لایه واحد اطلاعات بیت است. دراین لایه که ذاتاً سخت افزاری است، مسائل مخابراتی در مبادله بیت ها تجزیه و تحلیل شده و طراحی های لازم انجام می شود. طراحی شبکه می تواند برای پیاده سازی این لایه از استاندارد های شناخته شده ی انتقال همانند**RS-۲۳۲** ، **RS-۴۲۲** ، **RS-۴۲۳** استفاده کند.

لایه ی پیوند داده ها : وظیفه این لایه آن است که با استفاده از مکانیزم های کشف و کنترل خطا، داده ها را روی یک کانال انتقال که به دلیل پدیده ی اجتناب ناپذیر نویز ، ذاتاً دارای خطا است، بدون خطا و مطمئن به مقصد برساند. درحقیقت می توان وظیفه این لایه را بیمه اطلاعات در مقابل خطاهای احتمالی دانست. یکی دیگر از وظایف لایه ی پیوند داده ها آن است که اطلاعات ارسالی از لایه ی بالاتر را به واحد های استاندارد با طول مجاز تهیه کرده و پس از مشخص کردن ابتدا و انتهای آنها (از طریق نشانه های خاصی که **Delimiter** نامیده می شود) و افزودن فیلدهای اطلاعاتی خاص (مثل آدرس های مبدأ و مقصد ، کدهای کشف خطا و نظایرآن) یک فرم تشکیل داده ، سپس بیت های فریم سازماندهی شده را بیت به بیت جهت ارسال به لایه ی فیزیکی تزریق کند.

نکته : فریم عبارت است از یک قطعه اطلاعات دارای هویت که در لایه ی پیوند داده ها شکل می گیرد، در حقیقت فریم اسم خاص برای **PDU** ازلایه ی پیوند داده است. یکی دیگر از وظایف لایه ی دوم ،کنترل جریان یا به عبارت دیگر تنظیم جریان ارسال فریم ها به گونه ای است ، که یک گیرنده کند هیچگونه فریمی را به خاطر آهسته بودن پردازنده اش از دست ندهد.در این حالت فرستنده سریع باید سرعت ارسال فریم های خود را پایین بیاورد . کنترل جریان در لایه های بالاتر نیز وجود دارد ، ولی هرگاه در سطح این لایه نیز پیاده سازی شده باشد ، فرستنده و گیرنده نیاز به مقداری بافر خواهند داشت. مدیریت این بافر نیز بر عهده ی لایه ی پیوند داده ها است . یکی دیگر از وظایف این لایه می تواندآن باشد که وصول داده ها یا عدم دریافت صحیح هر فریم را به فرستنده اعلام کند.کلیه ی وظایف این لایه با استفاده ازسخت افزارهای دیجیتال انجام می شود.

لایه ی شبکه : اگر بخواهیم وظیفه این لایه را صرفاً در یک کلمه خلاصه کنیم ، آن کلمه مسیر یابی است، در این لایه داده های دریافتی از لایه ی بالاتر در قالب بسته هایی با ساختار استاندارد سازماندهی می شود و جهت انتقال بر روی خط خروجی تحویل سخت افزار لایه ی دوم می شود. با توجه به آنکه ممکن است بین دو ماشین در شبکه مسیرهای گوناگونی وجود داشته باشد ، لذا این لایه وظیفه دارد هر بسته ی اطلاعاتی را پس از دریافت به مسیری هدایت کند تا آن بسته بتواند به مقصد برسد. در این لایه باید تدابیری اندیشیده شود تا از ازدحام ، یعنی ترافیک بیش از اندازه ی بسته ها ، در یک مسیریاب یا سوئیچ جلوگیری شده و از ایجاد بن بست ممانعت به عمل آید. در این لایه تمام ماشین های شبکه نیازمند یک آدرس جهانی یکتا هستند و هر مسیریاب بر اساس این آدرس ها اقدام به هدایت بسته به سمت مقصد خواهد کرد، هر چند وظایف این لایه می تواند به صورت نرم افزاری

پیاده سازی شود ، ولی برای بالا رفتن سرعت عمل شبکه ، می توان برای این لایه سخت افزارخاص طراحی نمود تا بسته ها را با سرعت بسیار بالاتری بر روی شبکه رد و بدل کرد.

لایه ی انتقال : وظیفه کلیدی این لایه آن است که داده ها را از لایه ی بالاتر دریافت کرده ، در صورت نیاز آن را به قطعاتی با اندازه ی متناسب تقسیم بندی کرده و پس از ضمیمه کردن شناسه های لازم ، آن ها را جهت ارسال به لایه ی شبکه تحویل دهد.

خلاصه ی وظایف لایه ی انتقال را می توان موارد زیر برشمرد:

۱) کنترل جریان هوشمند در سطح پروسه (نه در سطح یک ماشین)

۲) کنترل مجدد خطا برای اطمینان نهایی

۳) تقسیم پیام های بزرگ به بسته های اطلاعاتی کوچکتر

۴) شماره گذاری بسته های قطعه قطعه شده جهت بازسازی

۵) بازسازی بسته های اطلاعاتی و تشکیل یک پیام کامل

۶) تعیین مکانیزم نام گذاری ایستگاه هایی که در شبکه اند

۷) حفظ ترتیب جریان بایت ها(باید به همان ترتیبی که دریافت شده اند به مقصد تحویل داده شوند)

در ادامه توضیحات لایه ی انتقال :پیاده سازی خدمات این لایه و لایه های بالاتر در سطح نرم افزار انجام می شود و قطعاً بر روی ماشین های نهایی (ماشین های کاربران) وجود دارد ، ولی در سوئیچ ها و مسیر یاب ها تمام بسته هایی که جهت هدایت بر روی مسیر مناسب داخل شده اند، قبل از آنکه وارد لایه ی چهارم شوند (یعنی در همان لایه ی سوم) ماشین را ترک می کنند.

لایه ی نشست: به مجموعه عملیاتی که پس از برقراری یک ارتباط بین دو پروسه ، با یک توافق اولیه آغاز ، سپس با انجام یک سری تراکنش ادامه می یابد و نهایتاً در روالی هماهنگ و مورد توافق ختم می شود، یک نشست می گویند.توافق اولیه می تواند شامل مجموعه ای از عملیات مفصل و پیچیده که شامل احراز هویت، صدور مجوز، مبادله گواهینامه ی دیجیتالی ، ایجاد رکوردهای حالت ، هماهنگی در خصوص شکل و ماهیت پیام ها و عملیاتی نظیر این باشد. لایه ی نشست تمام تمهیدات لازم برای ایجاد مدیریت و نگهداری نشست را فراهم می آورد و توانایی از سر گیری یک نشست نافرجام که به هر دلیل از جمله اشکال در لایه های زیرین یا قطع موقت ارتباط در شبکه ، ناتمام مانده است را دارد. در این لایه واحد اطلاعات پیام است.

وظایف کلی لایه ی نشست عبارتند از :

۱- برقراری و مدیریت یک نشست

۲- شناسایی طرفین

۳- سنکرون نمودن تماسها و فعل و انفعالات همزمان

۴- مشخص نمودن اعتبار پیامها

۵- اتمام نشست

۶- حسابداری مشتریان **(Accounting)**

لایه ی نمایش یا ارائه: عملیاتی که در این لایه انجام می گیرد ، عموماً بر روی محتوا و مفهوم پیام ها متمرکز است ، از این عملیات می توان به تبدیل قالب پیام ها ، فشرده سازی ، رمز نگاری و رمز گشایی پیام ها ، تبدیل کدها به یکدیگر مثلاً unicode و اسکی به یکدیگر اشاره نمود. هر گونه تغییر در ساختار محتوایی یا معنایی پیام در لایه ی نمایش انجام می گیرد.

لایه ی کاربردی: هر برنامه کاربردی که نهایتاً در خدمت کاربر نهایی شبکه قرار می گیرد در این لایه قرار دارد ، لذا این لایه را می توان مجموعه ای از استاندارد ها و پروتکل هایی دانست که برای تبادل پیام بین نرم افزار های کاربردی تعریف شده اند . به عنوان مثال قرارداد هایی نظیر پروتکل انتقال نامه های الکترونیکی ، پروتکل انتقال مطمئن فایل ، پروتکل دسترسی به بانک های اطلاعاتی راه دور ،پروتکل مدیریت شبکه ، پروتکل انتقال صفحات وب و صدها پروتکل دیگر در این لایه تعریف می شوند.

کیفیت خدمات یا Qos (Quality of service) : به دنباله ای از بسته ها که از یک پروسه در ماشین مبدأ تولید و به سوی یک برنامه کاربردی در ماشین مقصد روانه می شود جریان(flow) می گویند. بسته های متعلق به یک جریان ممکن است مسیر مشابهی را طی کنند، یا هر کدام از مسیری متفاوت به مقصد برسد ، بنابراین هر بسته ممکن است اتفاقات زیر را تجربه کند:

۱- بسته ها خارج ازترتیب و با تأخیرهای متفاوتی به مقصد برسند.
۲- میزان تأخیر هر بسته با دیگری اختلاف چشمگیری داشته باشد.
۳- برخی از بسته ها به دلیل بروز ازدحام در یک نقطه از مسیر هیچگاه به مقصد نرسند و در میانه راه حذف شوند.
۴- برخی از بسته ها به دلیل ماهیت پویای مسیریابی ، ناخودآگاه در مسیری قرار بگیرند که پهنای باند ناچیز و تأخیر ذاتی بالایی دارد.

نیازهای مطلوب هر جریان را می توان با ۴ پارامتر کمی زیر توصیف کرد :

تأخیر یا delay : میانگین کل زمانی که طول می کشد تا یک بسته پس از تولید در مبدأ تحویل گیرنده ی نهایی آن در مقصد شود است. در پارامتر تأخیر عواملی مثل تأخیر انتشار خط ، تأخیر صف بندی ، تعداد گام های مسیر و تأخیر Switching دخیل هستند.

لرزش یا jitter : عبارت است از انحراف معیار متغیر تصادفی تأخیر. لرزش را می توان نرخ اختلاف در زمان رسیدن بسته ها به مقصد تعبیر کرد. این اختلاف حول مقدار میانگین تأخیر سنجیده می شود ، به بیانی ساده پارامتر لرزش مشخص می کند که میزان تأخیر بسته های مختلف نسبت به مقدار متوسط تأخیر در چه محدوده ای تغییر دارد ، یا به اصطلاح می لرزد. کاربردهای صدا و تصویر شدیداً به لرزش حساس اند.

پهنای باند یا Bandwidth : عبارت است از نرخ متوسط تولید داده های یک جریان بر حسب بیت برثانیه به عبارتی دیگر نرخ داده های تولید شده توسط یک پروسه کاربردی در واحد زمان پهنای باند مورد نیاز آن تلقی می شود.

نرخ اتلاف بسته ها یا Paket Loss : میانگین از بین رفتن بسته های متعلق به یک جریان واحد ، که به دلایل متعدد در زیر ساخت شبکه اتفاق می افتد به نرخ اتلاف بسته شهرت دارد.

مثال هایی ازکاربرد های مشهور دنیا و نیازهای QOS آن ها عبارتند از:

پست الکترونیکی: ماهیت پست الکترونیکی را عموماً به صورت **offline** تصور می کنیم که نه به تأخیر و نه به لرزش حساس است و به پهنای باند کمی احتیاج دارد ، ولی در عوض باید قابلیت اطمینان صد در صدی داشته باشد یعنی شبکه باید اتلاف صفر مطلق را تضمین کند.

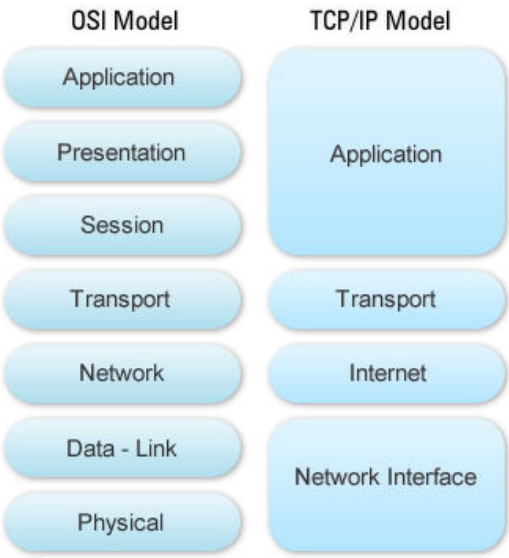
ftp : این کاربرد به تأخیر و لرزش حساسیت ندارد اما به پهنای باند متوسط به بالا نیازمند است و اتلاف باید صفر مطلق باشد.

websurfing : این کاربرد به لرزش حساس نیست ولی برای جلب رضایت کاربران شبکه باید ، تأخیر متوسطی داشته باشد. گردش در وب به پهنای باند متوسط و اتلاف صفر مطلق نیاز دارد.

دریافت صوت برحسب تقاضا (Audio On Demand): این کاربرد به تأخیر با مقدار ثابت چندان حساس نیست چرا که نواخته شدن یک فایل صوتی مثلاً با ۵ ثانیه تأخیر نسبت به مبداء زمان ایرادی ندارد ولی در عوض به لرزش حساس است ، چرا که اگر بسته های حاوی صدا ، دیرتر از زمان پیش بینی شده برسند ، صدای دریافتی قطع و وصل خواهد شد. این کاربرد به پهنای باند متوسطی نیاز دارد و می تواند تا سطحی اتلاف احتمالی بسته ها را تحمل کند ، چرا که یک یا چند بیت خرابی داده ، منجر به اندکی نویز در صدا خواهد شد.

مدل TCP/IP (Transfer Control Protocol / Internet Protocol)

مدلTCP/IPیک الگوی۴ لایه ای برای شبکه عرضه کرده است :



لایه اول از مدل TCP/IP :لایه ی واسط شبکه

در این لایه استاندارد های سخت افزار ، نرم افزارهای راه انداز(Device Driver) و پروتکل های شبکه تعریف می شود، این لایه درگیر با مسائل فیزیکی، الکتریکی و مخابراتی کانال انتقال ، نوع کارت شبکه و راه اندازی های لازم برای نصب کارت شبکه است.

لایه دوم از مدل TCP/IP :لایه ی شبکه

این لایه وظیفه دارد بسته های اطلاعاتی را که از این به بعد آن ها را بسته های IP می نامیم ، روی شبکه هدایت کرده و از مبدأ تا مقصد به پیش ببرد ، در این لایه چندین پروتکل در کنار هم وظیفه مسیریابی و تحویل بسته های اطلاعاتی از مبدأ تا مقصد را انجام می دهند ، کلیدی ترین پروتکل در این لایه، پروتکلIP نام دارد. در این لایه یک واحد اطلاعاتی که بایستی تحویل مقصد شود Datagram نامیده می شود ، پروتکل IP می تواند یک Datagram را در قالب بسته های کوچک تری قطعه قطعه کرده و پس از اضافه کردن اطلاعات لازم برای باز سازی ، آن ها را روی شبکه ارسال کند. در این لایه انتقال داده بین مبدأ و مقصد به روش بدون اتصال خواهد بود و ارسال یک بسته IP ، روی شبکه عبور از مسیر خاصی را تضمین نمی کند ، یعنی اگر دو بسته ی متوالی برای یک مقصد یکسان ارسال شود هیچ تضمینی در به ترتیب رسیدن آنها وجود ندارد ، چون این دو بسته می توانند از مسیرهای متفاوت به سمت مقصد حرکت نمایند. در ضمن در این لایه پس از آنکه بسته ای روی یکی ازکانال های ارتباطی هدایت شد ، از سالم رسیدن یا نرسید آن به مقصد هیچ اطلاعی به دست نخواهد آمد ، چرا که در این لایه برای بسته های IP هیچ گونه پیغام دریافت یا عدم دریافت بین عناصر واقع بر روی مسیر رد و بدل نمی شود، بنابراین سرویسی که در این لایه ارائه می شود، نامطمئن است، و اگر به سرویس های مطمئن و اتصال گرا نیاز باشد ، درلایه ی بالاتر این نیاز تأمین خواهد شد.

لایه سوم از مدل TCP/IP :لایه ی انتقال

این لایه ، ارتباط ماشین های انتهایی را در شبکه برقرار می کند ، یعنی می تواند بر اساس سرویسی که لایه ی دوم ارائه می کند یک ارتباط اتصال گرا و مطمئن بین پروسه های کاربردی برقرارکند، البته در این لایه برای عملیاتی نظیر ارسال صوت و تصویر که سرعت مهم تر از دقت برخطا است ، سرویس های بدون اتصال سریع و نامطمئن نیز فراهم شده است.

لایه چهارم از مدل TCP/IP :لایه کاربرد

در این لایه براساس خدمات لایه های زیرین سرویس سطح بالایی، برای خلق برنامه های کاربردی ویژه و پیچیده ارائه می شود، این خدمات در قالب پروتکل های استانداردی مانند خدمات web و انتقال صفحات ابر متنی ، مدیریت پست الکترونیکی ، انتقال فایل یا شبیه سازی ترمینال به کاربر ارائه می شود.

بخش دوم:

لا یه ی واسط شبکه ازمدل TCP/IP

مختصری در مورد کانالهای انتقال: وظیفه ی سخت افزار مخابراتی در لایه ی واسط شبکه آن است که بدون توجه به نوع و محتوای داده ها ، بیت های داده را بر روی کانال فیزیکی منتقل کند. سخت افزار انتقال در این لایه ، بیشتر با مسائل مخابراتی و الکتریکی سر و کار دارد. سه جزء اصلی این سخت افزار ، عبارتند از: گیرنده ، فرستنده و کانال انتقال.

پهنای باند: پهنای باند هر کانال را می توان توانایی و ظرفیت آن در ارسال اطلاعات با نرخ بیت بر هر ثانیه تعریف کرد. بنابراین وقتی گفته می شود پهنای باند یک کانال یک مگابیت بر ثانیه است ،یعنی با سرعت بالاتر از یک مگابیت بر ثانیه نمی توان اطلاعات را سالم به مقصد رساند.

نرخ خطای بیت: معیار خطا در کانالهای انتقال ، احتمال بروز یک بیت خطا روی کانال تعریف می شود، یعنی احتمال آنکه در فرستنده بیت۱ ارسال و در گیرنده اشتباهاً بیت۰ (و بالعکس) آشکار سازی شود. میانگین تعداد بیتهایی که در حین انتقال از طریق یک کانال دچار خطا می شوند، به نرخ خطای بیت یا **BER** شهرت دارد.

مالتی پلکسینگ: با توجه به آنکه پهنای باند بعضی از کانالها بسیار زیاد است (مثل کانالهای ماهواره ای یا فیبر نوری)می توان کانال فیزیکی را بین چندین ایستگاه تقسیم کرد. این تقسیم باعث می شود که از یک کانال مشترک چندین ایستگاه استفاده کنند و هزینه های ارتباط کاهش یابد. به عمل تقسیم پهنای باند یک کانال بین چند ایستگاه عمل «مالتی پلکس»یا «تسهیم»گفته می شود.

تسهیم در حوزه فرکانس یا FDM :

در روش **FDM** با فرض آنکه حداکثر **N** ایستگاه در شبکه وجود داشته باشد،پهنای باند فرکانسی کانال به **N** باند مجزا تقسیم می شود . هر ایستگاه موظف است در یکی از این باندهای فرکانسی ارسال و دریافت داشته باشد و چون این باند فرکانسی به صورت ثابت ، متعلق به خودش خواهد بود ، هرگونه تصادم و تداخل سیگنال منتفی است.

تسهیم درحوزه زمان یا TDM :

در روش **TDM** ، واحد زمان به بازه های کوچکی به نام برش زمان تقسیم شده و هر ایستگاه مجاز است فقط در برش زمانی متعلق به خودش، اطلاعات را روی کانال بفرستد.

روشهای **FDM** و **TDM**، زمانی کارآمد و مفید خواهند بود که اولاً تعداد ایستگاهها ثابت ومحدود باشد. ثانیاً هر ایستگاه حجم ثابت و در عین حال دائمی ارسال داده بر روی کانال داشته باشد. در شبکه های کامپیوتری ایستگاهها از نظر تعداد ، نامشخص و زیادند ، و داده ها نیز «انفجاری» هستند. انفجاری بودن ترافیک بدین معناست که ایستگاه در لحظاتی، به صورت ناگهانی حجم انبوهی از فریمها را برای ارسال روی کانال تولید می کند و سپس متوقف شده و تا لحظات متمادی هیچ داده ای برای ارسال تولید نمی کند.

کابل : «کابل» رسانه ای مادی است که از طریق آن سیگنال حامل اطلاعات از دستگاه مبدأ به سوی دستگاه مقصد انتقال می یابد. کابلهای مخابراتی بسیار متنوع اند ولیکن آنچه که امروزه برای شبکه های کامپیوتری اهمیت دارد رده های مختلف «کابلهای زوجی بهم تابیده» ، انواع« فیبرهای نوری» و « امواج مایکروویو» است.

سیمهای زوجی بهم تابیده :

سیمهای زوجی بهم تابیده عموماً در سه نوع ذیل به بازار عرضه می شود:

الف)سیم زوجی بهم تابیده ی بدون زره یا UTP : ارزانترین و رایجترین کابل در دنیای شبکه های کامپیوتری ، **UTP** است. این کابل در رده های مختلف به بازارعرضه شده وپهنای باند هر رده با دیگری متفاوت است.

ب)سیم زوجی بهم تابیده ی دارای فویل یا ScTP : این نوع از کابل دربرگیرنده ی چهارجفت سیم بهم تابیده است که برای کاهش اثر نویزهای خارجی بر روی مجموعه ی این چهار جفت یک فویل نازک آلومینیومی یا مسی کشیده شده است.

ج)سیم زوجی بهم تابیده ی زره دار یا STP : این نوع از کابل شامل چهار جفت سیم بهم تابیده است که بر روی هر جفت یک فویل کشیده شده و نهایتاً مجموعه ی جفت سیمها توسط یک پوشش ژاکت گونه فلزی در مقابل نویزهای بیرونی حفاظت می شود.

کابلهایUTP در هفت رده ی (Cat1 الی Cat7) به بازار عرضه شده اند :

Cat5 : این کابل با پهنای باند مفید ۱۰۰MHz در سال۱۹۹۴ به منظور بکارگیری درشبکه ی اترنت سریع ۱۰۰Base -T استاندارد شد. با این کابل نرخ ارسال اترنت در باند پایه و با یک جفت سیم به **100Mbps** رسید. در سال ۱۹۹۸ این رده از کابل با تغییر مختصر و بدون افزایش پهنای باند با نام **Cat5E** (ردهء ۵ ارتقاء یافته) برای بکارگیری در اترنت گیگابیت استاندارد سازی شد.

Cat6 : پهنای باند این کابل۲۵۰ مگاهرتز است و در دو نوع **Cat۶** و **Cat۶E** به بازارعرضه می شود.

Cat7 : با پهنای باند ۶۰۰ مگاهرتز برای کاربرد های آتی استاندارد سازی شد.

فیبرهای نوری :

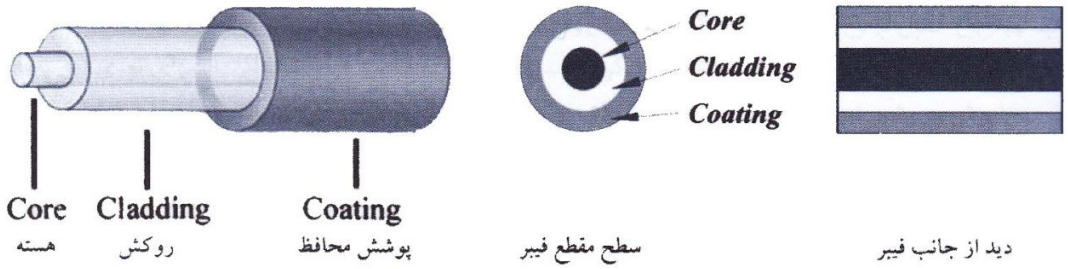
دریک عبارت ساده و غیرفنی«فیبرنوری»تارهای بسیار نازکی از جنس شیشه یا پلاستیک هستند که پرتوهای نور را از مبدأ به مقصد منتقل می کنند. فیبر نوری را یک استوانه ی به ضخامت یک تار مو و به طول چند متر تاچند ده کیلومتر تجسم کنید.

فیبر نوری از سه بخش تشکیل شده است:

۱)هسته: این بخش که در مرکز استوانه قرار دارد یک ماده بی نهایت شفاف از جنس شیشه یا پلاستیک است که پرتوهای نوری درون آن جریان می یابند. قطر این بخش بسته به نوع فیبر،چیزی بین۵ تا۵۰۰ میکرون است.

۲)روکش: این بخش که در پیرامون هسته ی مرکزی فیبر قرار دارد باز هم از جنس شیشه یا پلاستیک است با این تفاوت که ضریب شکست نور آن با هسته فرق می کند. در حقیقت ناحیه ی Cladding باعث خواهد شد که پرتو های نور تابانیده شده به درون هسته در اثر تفاوت ضریب شکست آنها با یکدیگر، همانند آینه منعکس شده و نتوانند از هسته خارج شوند.

۳)پوشش محافظ: این بخش که از جنس پلاستیک رنگی است برای محافظت فیبر در مقابل رطوبت یا خطرات فیزیکی برروی آن کشیده می شود. ضخامت پوشش محافظ بین۱۵۰ تا۱۰۰۰میکرون متغیراست. وقتی ابعاد یک فیبر به صورت ۵۰/۱۲۵ یا ۱۰/۱۲۵ میکرون معرفی می شود عدد اول ، قطر هسته و دیگری قطرخارجی روکش رامشخص کرده است.



ویژگیهای فیبرنوری:

۱- پهنای باند فوق العاده بالا
۲- ایمنی فوق العاده در مقابل نویز
۳- عدم ارتباط الکتریکی گیرنده و فرستنده
۴- امنیت اطلاعات
۵- وزن، حجم و قیمت پایین مواد اولیه
۶- ایمنی محیطی
۷- تضعیف ناچیز

خطا در شبکه های کامپیوتری:

خطا در خطوط انتقال جزو حقایقی است که به هیچ وجه نمی توان بطور کامل آن را برطرف کرد و همیشه جزو مشکلات عمده ی سیستمهای مخابراتی بوده و خواهد بود.

- ✓ (نویز حرارتی *شوک های الکتریکی *نویز کیهانی) از جمله عوامل بروز خطا هستند.

وقتی خطا در کانال های انتقال حقیقتی انکار ناپذیر است و از طرفی نیاز شبکه های کامپیوتری، انتقال عاری از خطا است، این تناقض باید به نحوی حل شود. برای این کار می توان به روش هایی متوسل شد تا گیرنده قبل از استفاده از داده ها سلامتی و بدون خطا بودن آنها را بر خود اثبات کند. در صورت وجود خطا ، داده ها دور ریخته می شود و ارسال مجدد صورت می گیرد.

روش کشف خطایChecksum : در این روش تمام بایت های یک فریم که باید توسط فرستنده ارسال شوند، با هم جمع (یا **XOR**) می شوند و یک بایت به نام «جمع کنترلی»یا **Checksum** بدست می آید. این بایت در انتهای فریم به مقصد ارسال می شود. در مقصد مجدداً بایت **Checksum** محاسبه و سپس مقایسه می شود. این روش در صورتی قادر به کشف خطا است که تعداد خطا های رخ داده در بیت های هم ارزش زوج نباشند.

10110001
01011101
01100101
10001001

10110001010111010110010 1 =>

مثال برای کشف خطا به روش **Checksum**

بایتها را با هم **XOR** نموده ایم و یک بایت برای جمع کنترلی بدست آورده ایم

کدهای کشف خطای CRC: در روشCRC به ازای مجموعه ی بیت های کل یک فریم ، تعدادی بیت کنترلی به نام کد **CRC** محاسبه و به انتهای فریم اضافه می شود. تعداد بیت های کد کشف خطای **CRC** مستقل از طول فریم و ثابت است. مبنای محاسبه ی کدهای **CRC** با استفاده از تقسیم چند جمله ای است که روش محاسبه ی آن با ارائه ی یک مثال توضیح داده شده است:

الف) ابتدا روی داده ی اصلی یک چند جمله ای تولید می شود. نمایش ریاضی چند جمله ای بدین صورت است که بیت ها از راست به چپ ضرایب یک چند جمله ای قرار می گیرند که توان هر جمله را موقعیت بیت در رشته مشخص می کند.

داده اصلی : 11100101

7	6	5	4	3	2	1	0
1	1	1	0	0	1	0	1

$$D(X) = 1*x^7 + 1*x^6 + 1*x^5 + 0*x^4 + 0*x^3 + 1*x^2 + 0*x^1 + 1*x^0$$

$$D(X) = x^7 + x^6 + x^5 + x^2 + 1$$

ب) بین گیرنده و فرستنده یک چند جمله ای به نام مولد قرار داده می شود.

$$G(x) = x^2+1$$

ج) برای تولید کدCRC، ابتدا چندجمله ای داده در جمله ی با بزرگترین توان مولد (**n**) ضرب می شود. سپس چند جمله ای $x^n.D(x)$ بر چند جمله ای مولد تقسیم می شود.(**n** بالاترین توان چندجمله ای مولد است) تقسیم در مبنای ۲ انجام می شود، یعنی ضرایب جملات با توان مساوی با هم **XOR** خواهند شد و تفریق معنا ندارد.

$$D(X).\ x^2 = x^9 + x^8 + x^7 + x^4 + x^2$$

$$x^9 + x^8 + x^7 + x^4 + x^2 \text{ Mod } G(x)$$

$$x^9 + x^8 + x^7 + x^4 + x^2 \ /\ x^2 + 1 = x^7 + x^6 + x^4 + 1$$

$x^9 + x^7$

$x^8 + x^4 + x^2$

$x^8 + x^6$

$x^6 + x^4 + x^2$

$x^6 + x^4$

x^2

$x^2 + 1$

باقیمانده = ۰۱

د) باقیمانده تقسیم $x^n.D(x)$ بر $G(X)$ با مقسوم جمع شده و نتیجه به عنوان داده ی جدید به شکل رشته بیت ارسال می گردد.

با این کار در حقیقت باقیمانده تقسیم به عنوان کدهای کنترل خطا در انتهای داده ها ارسال خواهند شد.

$x^9 + x^8 + x^7 + x^4 + x^2 + 1$ **11100101 01**

در روشی که بخواهیم به صورت باینری کد**CRC** را حساب کنیم، به تعداد بزرگترین توان جمله ی مولد ، در سمت راست رشته ی داده صفر اضافه می کنیم. بنابراین در مثال بالا باید دو صفر به سمت راست داده اضافه شود ، سپس داده ای که به آن صفر اضافه شده است، برچند جمله ای مولد تقسیم می شود. در تقسیم به نکات زیر باید توجه داشت:

۱)تقسیم ازاین لحاظ با تقسیم معمولی متفاوت است که شما باید از سمت چپ بیتهای باقیمانده را صفرکنید.

۲)جمع در مبنای۲ و به صورت**XOR** انجام می شود.

۳)نهایتاً بیتهای باقیمانده ی تقسیم، در سمت راست بیتهای داده قرار می گیرد.

مثال ۱)محاسبه کد کشف خطای **CRC** برای رشته ی ۱۱۱۰۰۱۰۱ و مولد ۱۰۱

```
1 1 1 0 0 1 0 1 0 0  | 1 0 1
1 0 1                | 1 1 0 1 0 0 0 1
  1 0 0
  1 0 1
    1 0 1
    1 0 1
      0 0 0 1 0 0
        1 0 1
          0 1  کد CRC
```

مثال ۲)محاسبه کد کشف خطای**CRC** برای رشته ی ۱۰۰۰۱۰۱ و مولد ۱۰۱

Data = $X^6 + X^2 + 1$

CRC Generator = $X^2 + 1$

```
1000101 00  | 101
101         | 1101001
 00101
  101
  0000100
     101
     01  کد CRC :
```

چند جمله ای های مولد زیر در دنیای شبکه های کامپیوتری از شهرت بیشتری برخوردارند :

CRC-12 $G(x) = x^{12}+x^{11}+x^3+x^2+1$

CRC-16 $G(x) = x^{16}+x^{15}+x^2+1$

CRC-CCITT $G(x) = x^{16}+x^{12}+x^5+1$

IEEE 802.x $G(x) = x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

چند جمله ای آخر برای تولید کد کشف خطای ۳۲ بیتی در شبکه های مشهور مثل اترنت و بیسیم **WiFi** کاربرد دارد.

استانداردهای انتقال روی خطوط نقطه به نقطه: بسیاری از کاربران اینترنت به وسیله مودم و از طریق خطوط تلفن معمولی به اینترنت متصل می شوند که کانالی نقطه به نقطه محسوب می شود.

پروتکل PPP : مهمترین پروتکل ارتباطی برای ایجاد یک لینک بین دو ماشین نقطه به نقطه می باشد. قبل از توضیح مشخصات دقیق فریم PPP ، ابتدا بررسی کنیم که وقتی از طریق یک خط سریال نقطه به نقطه (مثل خط تلفن) می خواهیم به اینترنت (یا یک ماشین راه دور) متصل شویم چه اتفاقاتی رخ می دهد تا یک اتصال موفقیت آمیز از نوع PPP برقرار شود.

1 Byte	1 Byte	1 Byte	1 or 2 byte	Variable	2 or 4	1 Byte
Flag 01111110	Address 11111111	Control 00000011	Protocol	Payload	Checksum	**Flag 01111110**

تنظیم و ایجاد یک اتصال PPP : در گام اول ماشین به کمک مودم شماره گیری می کند و پس از آنکه مودم طرف مقابل خط را وصل کرد و روی آن ، سیگنال حامل معتبر ظاهر شد، ابتدا لازم است یک سری بسته های کنترلی به نام LCP بین طرفین رد و بدل شود فریمهای LCP حاوی اطلاعات درون فیلد داده هستند که پارامترهای پروتکل PPP را به صورت توافقی ، انتخاب و تنظیم می نمایند در صورت موفقیت آمیز بودن گام اول ، در گام بعدی شروع کننده باید هویت خود را به طرف مقابل اثبات کند. سپس در گام سوم ،یک سری پارامترهای لایه ی بالاتر یعنی پروتکل لایه ی شبکه تنظیم می شود . این پارامترها توسط بسته هایی که NCP نام دارد تنظیم می شوند و طرفین باید بر روی این پارامتر ها مذاکره و توافق کنند. به عنوان مثال کامپیوتر شما که قبل از برقراری اتصال دارای آدرس IP نیست می تواند در هنگام برقراری اتصال با ISP از طریق رد و بدل کردن فریمهای NCP ،یک IP موقت بگیرد وآدرس IP خودش را تنظیم کند. پس از ختم اتصال، آن آدرس IP آزاد شده و می تواند به مشترک دیگری اختصاص داده شود. در ضمن در ابتدای برقراری اتصال بر روی طول فیلد داده ، طول فیلد کشف خطا، فیلد پروتکل و وجود یا عدم وجود فیلدهای آدرس و کنترل توافق می شود. روالی که به نهایی شدن این توافقات می انجامد ، مذاکره نام دارد. سرانجام مبادله ی فریم ها آغاز می شود ، در قسمت فیلد داده از پروتکل PPP ، می تواند بسته های IP یا بسته های هر پروتکل شناخته شده ی دیگر قرار بگیرد. حمل بسته های مورد نظر ادامه خواهد یافت تا طرفین بر سر ختم اتصال به توافق برسند. در هنگام ختم اتصال مجدداً بسته های NCP رد و بدل می شوند تا لایه ی بالاتر را از پایان اتصال مطلع کنند.سپس مجدداً یکسری فریمهای LCP مبادله می شود تا طرفین به صورت توافقی اتصال فیزیکی خودشان را قطع بنمایند و خط آزاد شود.

قالب فریم PPP : در قالب فریم PPP ، ابتدا وانتهای فریم با پرچم هشت بیتی(OX7E)۰۱۱۱۱۱۱۰ مشخص می شود و بالطبع چنین الگویی نباید در درون اطلاعات وجود داشته باشد .

فیلدهای بعدی فریم PPP به ترتیب عبارتند از :

۱) **Address Field:** این فیلد عملاً زائد است و در حین مذاکرات اولیه در مورد حذف آن از فریمهای آتی توافق میشود.

۲) **Control Field :** هر دو فیلد«آدرس» و«کنترل» ثابت وعملاً زائد هستند،که در شبکه ی اینترنت کاربردی ندارد.

۳) **Protocol :** عددی که در این فیلد قرار می گیرد، مشخص کننده ی آن است که بسته درون فیلد داده مربوط به چه پروتکلی در لایه ی بالاتر است، بدین معنا که پس از دریافت فریم ، محتوای فیلد داده ی آن باید برای پردازش های بعدی به کدام پروسه در لایه بالا تحویل شود (مثلاً پروتکل هایی مثل iP، iPX و.....) در ضمن این عدد می تواند مشخص کند که درون فیلد داده یک بسته NCP یا LCP قرار دارد.

۴) **Payload :** در این فیلد یک بسته ی مربوط به لایه ی بالاتر حمل می شود. محدودیتی بر روی طول این فیلد وجود ندارد ، ولی اندازه آن توافقی است و در ابتدای برقراری اتصال بین طرفین توافق می شود. در هنگام برقرای اتصال که هنوز طرفین اتصال روی اندازه مشخصی توافق نکرده اند، اندازه ی پیش فرض این فیلد ۱۵۰۰ بایت است.

۵) **Checksum:** این فیلد برای درج کد کشف خطای فریم است و در حالت پیش فرض، دو بایتی فرض می شود ، ولی طرفین اتصال می توانند در مرحله مذاکره آن را بصورت چهار بایتی توافق کنند.چند جمله ای های مولد برای محاسبه کد کشف خطا بصورت زیر قرار داده شده اند (ولی برای کدCRC با پیش فرض دو بایتی و دومی برای حالتی که بر روی کد چهار بایتی توافق می شود):

CRC-CCITT	$x^{16}+x^{12}+x^5+1$
IEEE 802.x	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

احراز هویت به کمک بسته های LCP: یکی از ویژگیهای بنیادی درPPP آنست که قبل از تنظیم نهایی لینک بین دو ماشین روبرو هر یک از طرفین می تواند هویت طرف مقابل خود را اثبات کند. احراز هویت درPPP به روشهایPAP وCHAP قابل انجام است.

در روش دو مرحله ایPAP : کسی که دارای کلمه ی عبور معتبری از همتای خود است باید پس از مذاکرات اولیه ، بلافاصله با ارسال بسته LCPAuthenticate Request شناسه (ID) و کلمه عبور خود را به طرف مقابل تسلیم کند. طرف مقابل با جستجوی آن در« پایگاه کلمات عبور صادر شده برای افراد» با ارسال بسته ی LCP Authenticate ACK طرف مقابل را تأیید و اجازه ادامه کار را می دهد و یا با ارسال بسته یLCP Authenticate NACK از پذیرش وی سر باز می زند و اتصال قطع می شود. این روش به دلیل آن که کلمه عبور بصورت آشکار روی خط منتقل می شود بسیار نا امن است چرا که امکان استراق سمع و سوء استفاده از آن فراهم است.

در روشCHAP فرآیند کار به شکل زیر است:

برای احراز هویت به کلید رمز نیاز است که از آن به عنوان کلمه عبور یاد می شود ولی در حقیقت کلید رمز نگاری و در هم سازی اطلاعات است. قرار نیست این کلمه عبور بر روی خط ارسال شود بلکه احراز هویت طبق روال زیر انجام می گیرد:

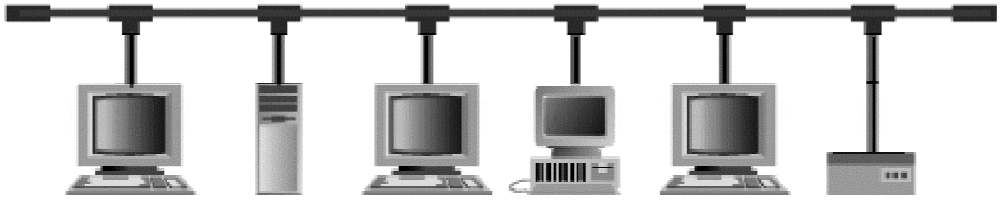
*هر یک از طرفین (در صورت تمایل)طرف مقابل را به چالش دعوت می کند ، این کار با ارسال یک عدد تصادفی۱۶ بایتی (مشهور به رشته ی چالش) انجام می گیرد.

*اگر طرف مقابل کلمه ی عبور درستی را در اختیار داشته باشد باید رشته چالش طرف مقابل را به کمک این اسم رمز طبق الگوریتمی خاص در هم سازی و رمز نگاری کند و حاصل را برای همتای خود پس بفرستد. بدین ترتیب چیزی که بر روی خط ارسال می شود هیچ نشانی از کلمه عبور در آن نیست و در دفعات مختلف تغییر خواهد کرد.

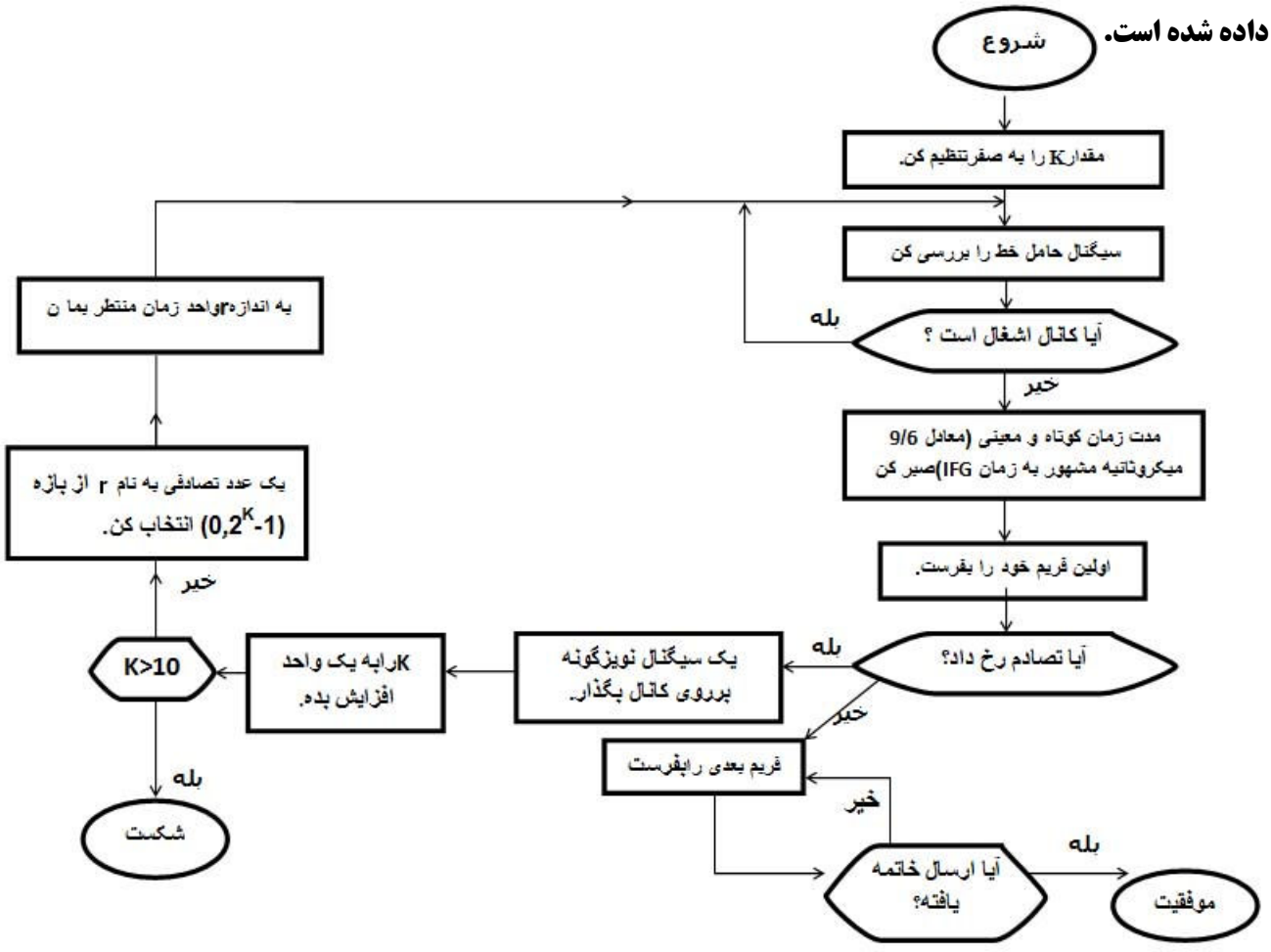
*طرف مقابل داده های ارسالی را به کمک پایگاه داده ی کلمات عبور ،کشف رمز کرده و نتیجه را با مقدار ارسالی در مرحله اول مقایسه می کند و اگر نتیجه مقایسه مثبت بود بسته‌ء LCP SUCCESS و در صورت منفی بودن، بسته‌ء LCP FAILURE را بر می گرداند.

اترنت سنتی (IEEE 802.3) :

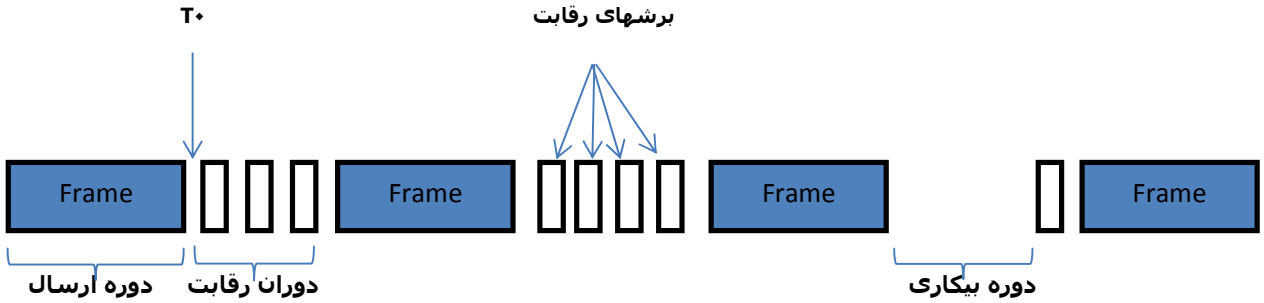
این استاندارد برای شبکه های کانال مشترک با توپولوژی باس تعریف شده است این استاندارد در سال ۱۹۸۳ به تصویب IEEE رسید تا سرعت اترنت را به ده گیگابیت بر ثانیه ارتقا بدهد در این بخش ابتدا اترنت سنتی با استاندارد IEEE ۸۰۲.۳ مبتنی بر توپولوژی باس را بررسی خواهیم کرد. در اترنت سنتی تمام ایستگاه ها از طریق یک کابل چند اتصالی به کانالی مشترک متصل می شوند



الگوریتم CSMA/CD یا دسترسی چند گانه با قابلیت شنود سیگنال حامل و کشف تصادم مشهور است در نمودار زیر نشان داده شده است.



مدل مفهومی شکل زیر برای CSMA/CD شامل دوره های زمانی زیر است:چندین مرحله ی متوالی رقابت (Contention) بازه ی ارسال فریم و بازه های بیکاری (زمان هایی که تمام ایستگاه ها به دلیل عدم نیاز به ارسال فریم ساکت بوده اند).روال CSMA/CD از مدل زیر تبعیت می کند:



در لحظه ای که با T0 مشخص شده ارسال فریم یک ایستگاه به پایان رسیده و خط آزاد شده ، در این لحظه ایستگاه هایی که فریمی برای ارسال آماده دارند ممکن است برای ارسال آن تلاش کنند. اگر دو یا چند ایستگاه بطور همزمان اقدام به ارسال نمایند تصادم رخ خواهد داد و ارسال تکرار خواهد شد. زمان کشف تصادم که مستقیماً به طول کانال وابسته است جزو زمانهای تلفاتی محسوب می شود که به ازای ارسال هر فریم ممکن است اتفاق بیفتد . گاهی مجموع این زمانهای تلفاتی از زمان لازم برای ارسال یک فریم فراتر می رود.کشف تصادم به صورت آنالوگ انجام می گیرد ، این کار به سه روش ((بررسی توان مصرفی)) ، ((اندازه گیری پهنای پالس سیگنال دریافتی از کانال و مقایسه آن با پهنای واقعی سیگنال ارسالی)) و ((اندازه گیری سطوح ولتاژ پالسها)) قابل انجام است.

۴–راندمان کانال در این استاندارد به پارامترهای زیر بستگی دارد:

- F : طول فریم بر حسب بیت
- C : سرعت انتشار
- B : پهنای باند کانال
- L : طول کانال
- E : عدد نپر (2.718281~)

$$\text{راندمان کانال مشترک} = \frac{1}{1+\frac{2.e.B.L}{C.F}}$$

با دقت در این رابطه مشهود است که:

۱) با کاهش طول فریم(f)راندمان کانال هم کاهش می یابد ، زیرا زمان تلف شده برای رقابت و تصرف کانال بر روی حجم کمی از داده سر شکن می شود.

۲) با افزایش طول کانال(L)راندمان کانال کاهش می یابد ، زیرا زمان تلفاتی برای رقابت و تصرف کانال به نسبت طول کانال افزایش می یابد.

۳) با افزایش نرخ ارسال راندمان کانال کاهش می یابد زیرا در زمان تلف شده برای رقابت و تصرف کانال،حجم داده ی بیشتری از بین می رود.

نکته۱ :در استاندارد اترنت دو نوع کابل به عنوان رسانه انتقال مجاز شمرده شد : اول کابل کواکیسال ضخیم(۱۰Base۵)که دراین نوع کابل کشی کامپیوترها از طریق یک سیم چند رشته ای به دستگاه کوچکی به نام ترانسیور متصل می شوند. کابل(۱۰Base۵)که به اترنت ضخیم مشهور شد، به رنگ زرد و شبیه شیلنگ باغبانی بود و هر۲/۵مترعلامتی بر روی کابل برای مشخص کردن محل مناسب برای انشعاب تزریقی ،گذاشته شده بود. نوع دوم کابل اترنت (۱۰Base۲)نام داشت که به کابل اترنت نازک مشهور شد ، این نوع کابل به دلیل نازکی قابل انعطاف بود و به راحتی می شد به جای استفاده از انشعاب تزریقی ،آن را از طریق کانکتورBNC ، مستقیماً به کارت شبکه متصل کرد ، بدین ترتیب مدار ترانسیور بر روی کارت شبکه تعبیه شد.

نکته۲ : تفاوتی که کابل کشی با کابلهای ضخیم یا نازک داشت، این بود که در صورت استفاده از کابل ضخیم حداکثر طول یک قطعه کابل بدون نیاز به تکرار کننده تا ۵۰۰ متر قابل گسترش بود و به هر قطعه ی ۵۰۰ متری می شد تا۱۰۰ ایستگاه متصل کرد، در حالی که با کابل نازک حداکثر طول یک قطعه کابل به ۱۸۵ متر و تعداد ایستگاه های قابل اتصال به این قطعه به۳۰ دستگاه کاهش می یافت.

نکته۳: کارت شبکه اترنت NIC (Network Interface Card) دارای یک کنترلر خاص است که وظیفه دارد داده ها را در قالب یک فریم مناسب سازمان دهی کند، همچنین محاسبه کدهای کشف خطا برای فریم های خروجی و ارزیابی صحت فریم های ورودی بر عهده ی همین کنترلر است، این کنترلر دارای مقداری حافظه RAM است ، تا بتواند یک یا چند فریم داده را به صف و بافر نماید. در ضمن این کنترل قادر است داده ها را به روش DMA به حافظه اصلی کامپیوتر منتقل نماید.

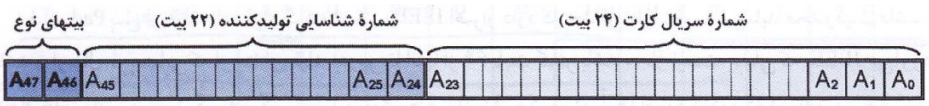
نکته۴: سرعت انتقال در اولین نسخه ی اترنت استاندارد ، ۱۰مگابیت بر ثانیه است و برای ارسال بیت ها از روش کدینگ منچستر استفاده شده است.

قالب فریم پیشنهادی IEEE برای اترنت :

Bytes	7	1	6	6	2	0-1500	0-46	4
	Preamble	SOF	Destination address	Source address	Length	Data	Pad	Check-sum

هر فریم با هفت بایت دیباچه(preamble)آغاز می شودکه تمام بایت ها دارای الگوی ۱۰۱۰۱۰۱۰ هستند،کدینگ این الگوی۷ بایتی، به مدت ۵/۶ میکرو ثانیه،یک سیگنال ساعت مربعی ده مگاهرتز تولید می کند ،تا به کمک آن تمام ایستگاه های گیرنده بتوانند سیگنال ساعت خود را با سیگنال ساعت فرستنده سنکرون کنند، در ادامه فیلد یک بایتی SoF (Start of Frame) آمده است، که دارای الگوی ۱۰۱۰۱۰۱۱ است. فیلد Pad ، پس از فیلد داده قرار گرفته است ، IEEE اصرار دارد که حداقل طول فریم باید محدودیت داشته باشد و ارسال فریم هایی که از ابتدای فیلد آدرس تا انتها از ۶۴ بایت کمتر باشد، مجاز نیست. اگر فریمی از ۶۴ بایت کمتر شود(بدون احتساب فیلد Preamble و SoF) باید در فیلد Pad ، آنقدر داده های زائد اضافه شود تا طول آن به ۶۴ بایت برسد از آنجایی که طول داده های مفید در فیلد Length دقیقاً مشخص است، لذا اضافه کردن داده های زائد مشکلی را ایجاد نخواهد کرد. کنترلر کارت شبکه به صورت بی درنگ برای مجموعه بیت های در حال ارسال از ابتدای فیلد آدرس مقصد تا انتهای فیلد داده یک کد ۳۲ بیتی CRC، محاسبه و استخراج کرده و آنرا در فیلد ۴ بایتی Checksum جایگذاری می نماید.

آدرس ها در اترنت :



هر کارت شبکه اترنت دارای یک آدرس سخت افزاری یکتا است که در درون **ROM** ذخیره شده و هویت یک ایستگاه را مشخص می نماید به این آدرس ، آدرس **MAC** یا آدرس فیزیکی نیز گفته می شود. هرگاه ایستگاهی اقدام به ارسال فریمی نماید در فیلد آدرس مبدأ از فریم ، آدرس سخت افزاری آن درج می شود آدرس فیزیکی ۴۸ بیتی است و از سه بخش تشکیل شده است. بیت **A۴۶** سراسری یا محلی بودن آدرس را مشخص می کند. اگر این بیت صفر باشد، بدن معناست که این آدرس توسط مدیر شبکه محلی تعیین شده و در خارج از شبکه هیچ ارزشی ندارد، اگر این بیت برابر یک باشد، بیانگر آن است که این آدرس توسط **IEEE** به ثبت رسیده و اعتبار جهانی دارد.

در اترنت یک فریم را می توان با سه نوع آدرس ارسال کرد :

آدرس تک پخشی : بدین معناست که گیرنده فریم یک استگاه واحد است، در این حالت پر ارزش ترین بیت(**A47**) برابر صفر است.

آدرس های چند پخشی : بدین معنا که گیرنده فریم یک گروه خاص از ایستگاه ها هستند، در این حالت پر ارزش ترین بیت برابر یک است و مابقی بیت ها شماره ی گروه رامشخص می کنند.

آدرس پخش فراگیر: به این معنا که گیرنده فریم تمام ایستگاه های متصل به کانال خواهند بود، اگر تمام بیت های فیلدآدرس مقصد برابر ۱ باشند تمام ایستگاه ها فریم را از روی کانال دریافت کرده و آن را پردازش خواهند کرد.

به عبارت دیگر یک ایستگاه در شبکه اترنت در سه حالت ممکن است فریمی را از روی کانال برداشته و پردازش کند:

۱- در فیلد آدرس مقصد در فریم ، آدرس سخت افزاری خود را ببیند.

۲- در فیلد آدرس مقصد شماره گروهی را ببینید که او هم در آن گروه عضو است.

۳- در فیلد آدرس مقصد تمام بیت ها برابر۱ باشند.

ظهور هاب در استاندارد IEEE802.3:

درسال۱۹۹۱ **IEEE** روش جدیدی برای کابل کشی اترنت به نام **10BaseT** ارائه کرد ، که در آن کانال مشترک در درون جعبه ای به نام(**HUB**) تعبیه شده و تمام ایستگاه ها توسط یک کابل اختصاصی به آن متصل می شوند. در توپولوژی ستاره تمام ایستگاه ها با سیم های زوجی از نوع **Cat۵** به هم متصل می شوند با توجه به محدودیت سیم های زوجی حداکثر فاصله ایستگاه ها تا هاب نباید از ۱۰۰ متر تجاوز میکرد.

هاب را می توان به دو صورت پیاده سازی کرد :

الف) هاب غیر فعال :

یک جعبه پر از سیم است که تنها وظیفه اش به هم وصل کردن کابل ایستگاه ها به یکدیگر و ایجاد کانالی مشترک است، این هاب به منبع تغذیه نیازی ندارد و هیچگونه کمکی به کشف تصادم یا تقویت سیگنال نمی کند و در یک کلمه هوشمند نیست.

ب) هاب فعال :

هاب فعال گذشته از برقراری اتصال ایستگاه ها به یک باس مشترک ، عملیات زیر را نیز انجام می دهد:

۱- سیگنال ورودی ایستگاه ها را قبل از اعمال بر روی کانال مشترک تقویت و بازتولید می کند.

۲- عملیات کشف تصادم(Collision Detection) و تولید سیگنال نویزگونه JAM (جهت اطلاع به کل ایستگاهها در خصوص بروز تصادم) بر عهده ی هاب گذاشته شده است.

۳- بخشی از عملیات ترانسیور که در ساختار ۱۰Base۵ مورد استفاده قرار می گرفت ، در ساختار جدید بر عهده ی هاب گذاشته شده است ، یعنی خطوط ارسال و دریافت داده ها به طور جدا و مستقل وارد هاب می شوند، هاب سیگنال دریافتی را پس از باز تولید و تقویت بر روی کانال مشترک منتقل می کند سپس آنچه را که بر روی باس مشترک جریان دارد را(در صورت عدم تصادم) به طور جداگانه بر روی خطوط دریافت(Recieve) تمام ایستگاه ها باز تولید و تکرار می کند.

۴- هرگونه خرابی و عملکرد غیر صحیح یک ایستگاه منجر به از کار افتادن کل شبکه نخواهد شد چرا که هاب هوشمندانه ایستگاه خراب را از مدار خارج خواهد کرد. هاب فعال تنها سیگنال های حامل داده ی معتبر و مطابق با استاندارد را بر روی کانال مشترک می فرستد ، طبیعی است که هاب فعال برای انجام وظیفه به منبع تغذیه مستقل نیازمند است.

نام کابل	نوع کابل	حداکثر طول قطعه	تعدادگره در هر قطعه	مزایا
10Base5	Thick coax	500m	100	کابل اصلی و اولیه (از رده خارج)
10Base2	Thin coax	185m	30	به هاب نیازی نیست
10Base-T	Twisted pair	100m	1024	ارزان ترین سیستم
10Base-F	Fiber optics	2000m	1024	بهترین انتخاب برای مابین ساختمانها

نکات :

۱- در اترنت سنتی تمام ایستگاه هایی که به یک کانال مشترک گوش می دهند. اصطلاحاً بر روی یک حوزه ی پخش فراگیر(Broadcast Domain) واقع هستند. وقتی گفته می شود حوزه ی تصادم (مثلاً) ۱۶ است یعنی ۱۶ ایستگاه برای دسترسی به کانال مشترک با هم رقابت می کنند.

۲- تکرار کننده (Repeater) ابزاری است آنالوگ که دو قطعه کابل را بهم پیوند می زند سیگنالی که بر روی یکی از این قطعات ظاهر گردد پس از دریافت و تشخیص بیت «بازتولید» و تقویت شده و بر روی قطعه دیگر قرار داده می شود.

۳- حداکثر طول کابل بدون آنکه نیازی به تکرار کننده باشد ، یک قطعه (segment) نام دارد، در شبکه اترنت حداکثر می توان ۵ قطعه کابل را به کمک ۴ تکرار کننده بهم متصل کرد و از این ۵ قطعه فقط می توان به ۳ قطعه کابل استگاه متصل کرد و از دو قطعه کابل دیگر فقط برای ارتباط و گسترش طول کانال(در صورت نیاز) استفاده می شود ، به چنین محدودیتی (قاعد۳-۴-۵) گفته می شود.

۴- در اترنت دریافت فریم ها توسط گیرنده تأیید نمی شود یعنی پس از ارسال موفق یک فریم ، سخت افزار گیرنده دریافت آن را گزارش نمی کند و برگرداندن پیغام تصدیق بر عهده لایه های بالاتر گذاشته شده است.

بخش سوم

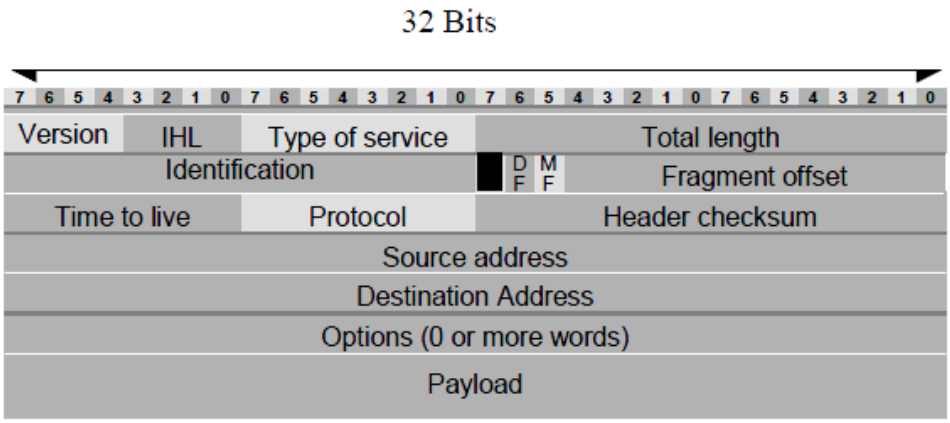
لایه ی IP در شبکه ی اینترنت

تعریف پروتکل IP :

قراردادی که حمل و تردد بسته های اطلاعاتی و همچنین مسیریابی صحیح آن ها ازمبدأ به مقصد را مدیریت و سازماندهی می نماید پروتکلIP نام دارد. هدایت بسته ها توسط پروتکل IP بدین صورت انجام می شود که ابتدا لایه ی IP یک واحد از داده ها به نام دیتاگرام را از لایه ی بالاتر تحویل گرفته و در صورت بزرگ بودن آن را به واحدهای کوچکتری به نام قطعه شکسته و با تشکیل یک بستهIP به ازای هرقطعه ، اطلاعات لازم برای طی مسیر در شبکه را به آن ها اضافه می کند ، سپس آنها را روی شبکه به جریان می اندازد. هر مسیریاب با بررسی و پردازش بسته ها ، آن ها را تا مقصد هدایت می کند هرچند طول یک بسته می تواند حداکثر ۶۴ کیلوبایت باشد ولی در عمل عموماً طول بسته ها حدود ۱۵۰۰ بایت است. پروتکل IP مجبور است هنگام قطعه قطعه کردن یک دیتاگرام برای کل آن یک شماره مشخصه و برای هر قطعه یک شماره ترتیب در نظر بگیرد تا آن دیتاگرام بتواند در مقصد برای تحویل به لایه ی بالاتر یعنی لایه ی انتقال بازسازی شود.

قالب یک بسته IP :

یک بسته IP از دو قسمت سر آیند (Header) و قسمت حمل داده (Payload) تشکیل شده است. مجموعه اطلاعاتی که در سرآیند بسته ی IP درج می شود ، توسط مسیر یاب ها مورد استفاده و پردازش قرار می گیرد.



فیلدVersion: اولین فیلد چهار بیتی سر آیند یک بستهIP نسخه ی پروتکل IP را(که این بسته بر اساس آن، سازماندهی و ارسال شده) تعیین می کند،تمام شبکه ها و مسیریاب ها ، از نسخه شماره ی ۴ پروتکلIP پشتیبانی می کنند، بنابراین در این فیلد، مقدار باینری۰۱۰۰ قرار می گیرد.

فیلد IHL(IPHeaderLenght) : این فیلد هم چهار بیتی است ، و طول کل سرآیند بسته را بر مبنای کلمات ۳۲ بیتی مشخص می نماید به عنوان مثال اگر در این فیلد عدد۱۰ قرار گرفته باشد ، بدین معناست که کل سرآیند ۳۲۰ بیت ، معادل چهل بایت خواهد بود، اگر به ساختار یک بسته ی IP دقت شود به غیر از فیلد Option که اختیاری است ، وجود تمام فیلد های سر آیند الزامی است. از آنجا که طول قسمت اجباری سرآیند۲۰ بایت است ، حداقل عددی که در فیلدIHL قرار می گیرد، باید برابر۵(۰۱۰۱) باشد و هر مقدار کمتر از۵ به عنوان خطا تلقی و منجر به حذف بسته خواهد شد. با توجه به طول۴ بیتی این فیلد، بدیهی است که حداکثر مقدار آن برابر۱۵ (۱۱۱۱) است که در این صورت، طول قسمت سر آیند، ۶۰ بایت و فضای باقی مانده برای بخش اختیاری برابر۴۰ بایت خواهد شد.

فیلد Type of Service : این فیلد ۸ بیتی است و توسط آن ماشین میزبان یعنی ماشین تولید کنندهٔ ی بسته ی IP ازمجموعه ی زیر شبکه ، یعنی مجموعه ی مسیر یاب های بین راه تقاضای سرویس ویژه ای برای ارسال یک دیتا گرام می نماید. به عنوان مثال ممکن است یک ماشین میزبان بخواهد دیتاگرام صدا یا تصویر برای ماشین مقصد ارسال نماید ، در چنین شرایطی از زیر شبکه ، تقاضای ارسال سریع و به موقع اطلاعات را دارد ، نه قابلیت اطمینان۱۰۰ ٪ . برای مشخص کردن ماهیت سرویس مورد تقاضا ، این فیلد به چند زیر فیلد تقسیم شده است.

P2	P1	P0	D	T	R	-	-
تقدم بسته			تأخیر	ظرفیت خروجی	قابلیت اطمینان	بی استفاده	

الف) بیت های P۰ ، P۱ ، P۲ : این سه بیت در سمت چپ ، اولویت بسته ی IP را تعیین می کند. اگر در این سه بیت ، صفر قرار گرفته باشد ، بسته ی اطلاعاتی از نوع معمولی تلقی می شود، یعنی دارای پایین ترین مقدار اولویت است و مقدار۷ در این سه بیت قرارگرفته باشد، بالاترین اولویت ، برای بسته در نظر گرفته می شود ، مسیریاب در بین بسته های IP که ازکانال های مختلف وارد شده اند ، بسته هایی را زودتر پردازش و مسیر یابی می کند دارای اولویت بالاتری باشند.

ب) بیت های R ، T ، D : بیت D به معنای تأخیر Delay ، بیت R به معنای قابلیت اطمینان (Reliability) و بیت T به معنای ظرفیت خروجی خط (Throughput) است و ماشین میزبان با قرار دادن ۱ در این بیت ها،انتظارش را از زیرشبکه بیان می کند.

فیلد Total Length : در این فیلد۱۶ بیتی، عددی قرار می گیرد که طول کل بسته ی IP را که شامل مجموع اندازه سر آیند و ناحیه ی داده است تعیین می کند، مبنای طول بر حسب بایت است و بنابراین حداکثر طول کل بسته یIP می تواند۶۴ کیلوبایت باشد ولی در عمل اینگونه نیست و اندازه ی اغلب بسته ها کمتر از۱۵۰۰ بایت است.

فیلد Identification : وقتی دیتاگرام شکسته می شود، باید شناسه ای داشته باشد تا در هنگام بازسازی آن درمقصد ، بتوان قطعه های آن دیتاگرام را از بقیه جدا و سرهم کرد، در این فیلد۱۶ بیتی ،عددی قرار می گیرد که شماره ی یک دیتاگرام را مشخص می کند.

فیلد Fragment Offset: این فیلد خود از سه زیر فیلد تشکیل شده است:

الف) بیت DF(Don’t Fragment) : با یک شدن این بیت در یک بسته IP ، هیچ مسیریابی حق ندارد آن را قطعه قطعه کند، چرا که مقصد به هر دلیل قادر به بازسازی دیتاگرام های تکه تکه شده نیست .

ب) بیت MF(More Fragment): این بیت مشخص می کند که آیا بسته IP آخرین قطعه از یک دیتاگرام محسوب می شود یا باز هم قطعه های بعدی وجود دارد ، در آخرین قطعه از یک دیتا گرام بیت MF صفر خواهد بود و در بقیه قطعات الزاماً برابر ۱ است .

ج) Fragment Offset : این زیر فیلد ۱۳ بیتی ، شماره ترتیب هر قطعه در دیتا گرام شکسته شده محسوب می شود ، با توجه به ۱۳ بیتی بودن این زیرفیلد یک دیتاگرام حداکثر می تواند به ۸۱۹۲ تکه تقسیم شود ، نکته مهم در مورد این فیلد آنست که اندازه هر قطعه باید ضریبی از ۸ باشد به استثنای قطعه آخر، به عنوان مثال فرض کنید مسیریابی مجبور است که یک دیتاگرام به طول۵۰۰۰ بایت را قطعه قطعه کند به گونه ای که اندازه هر قطعه کمتر از ۱۵۰۰ بایت باشد در چنین موردی نمی تواند اندازه هر قطعه را ۱۲۵۰ بایت در نظر بگیرد ، چرا که ضریبی از ۸ نیست ولی اندازه ۱۲۸۰ مناسب است ، در این حالت مسیر یاب دیتاگرام را به سه بسته ۱۲۸۰ بایتی و یک بسته ۱۱۶۰ بایتی می شکند.

شمارهٔ قطعه	Identification	Fragment Offset	بیت MF	آدرس محل قرار گرفتن قطعه در دیتاگرام	طول هر قطعه
قطعهٔ شماره ۱	2322	0	1	8*0=0	۱۲۸۰
قطعهٔ شماره ۲	2322	160	1	8*160=1280	۱۲۸۰
قطعهٔ شماره ۳	2322	320	1	8*320=2560	۱۲۸۰
آخرین قطعه	2322	480	0	8*480=3840	۱۱۶۰

فیلد TTL(Time To Live): این فیلد ۸ بیتی در نقش یک شمارنده ، طول عمر بسته را مشخص می کند. طول عمر یک بسته به طور ضمنی به زمانی اشاره می کند که یک بسته IP می تواند برروی شبکه سرگردان باشد ، حداکثر طول عمر یک بسته برابر ۲۵۵ خواهد بود که به ازای عبور از هر مسیریاب از مقدار این فیلد یک واحد کم می شود، و اگر به دلیل بافر شدن در حافظه یک مسیر یاب زمانی را معطل بماند ، به ازای هر واحد زمانی یک واحد از این فیلد کم خواهد شد. به محض آنکه مقدار این فیلد به صفر برسد بسته IP در هر نقطه از مسیر که باشد ، بلافاصله حذف شده و از ادامه مسیر آن به سمت مقصد جلوگیری خواهد شد ، البته معمولاً یک پیام هشدار به ماشینی که آن بسته را تولید کرده باز پس فرستاده خواهد شد.

نکته: اگرچه بزرگترین عددی که در فیلد طول عمر بسته قرار می گیرد برابر ۲۵۵ است ولی ، در عمل مقداری که سیستم های عامل در این فیلد قرار می دهند عددی بین ۳۲ تا ۱۲۸ است. مسئول شبکه می تواند مقدار پیش فرض TTL را در سیستم عامل عوض کند.

فیلد پروتکل: مقدار این فیلد ، شماره ی پروتکلی است که در لایه ی بالاتر تقاضای ارسال یک دیتاگرام کرده است ، هر یک از پروتکل های لایه ی بالاتر دارای یک شماره ۸ بیتی منحصر بفرد و استاندارد هستند.

فیلد Header Checksum : این فیلد ۱۶ بیتی به منظور کشف خطاهای احتمالی در سرآیند هر بسته IP استفاده می شود. برای محاسبه کد کشف خطا ، کل سرآیند به صورت دو بایت دو بایت با یکدیگر جمع می شود، نهایتاً حاصل جمع به روش مکمل یک منفی و نتیجه در این فیلد از سرآیند قرار می گیرد. هر مسیریاب قبل از پردازش و مسیر یابی ابتدا صحت اطلاعات درون سرآیند را بررسی می کند. روش بررسی بدین صورت است که اگر تمام سرآیند به صورت دوبایت دوبایت در مبنای مکمل یک با یکدیگر جمع شود باید حاصل جمع صفر بدست آید ، در غیر این صورت بستهIP فاقد اعتبار بوده و حذف خواهد شد. دقت کنید که فیلد checksum در هر مسیریاب باید از نو ، محاسبه و مقدار دهی شود زیرا وقتی یک بسته IP وارد مسیریاب می شود حداقل فیلدTTL از آن بسته عوض خواهد شد. فیلد Checksum برای کشف خطاهای احتمالی درون داده های فیلد payload استفاده نمی شود زیرا اینگونه خطاها در لایه ی پایین تر یعنی لا یه ی فیزیکی معمولاً توسط کدهای CRC نظارت خواهند شد.

فیلد Source Address : در پروتکل نسخه چهار IP ، هر ماشین میزبان در شبکه اینترنت یک آدرس جهانی و یکتای ۳۲ بیتی دارد. بنابراین هر ماشین میزبان در هنگام تولید یک بسته IP باید آدرس خودش را در این فیلد قرار بدهد.

فیلد Destination Address : در این فیلد آدرس ۳۲ بیتی مربوط به ماشین مقصد که باید بسته ی IP را تحویل بگیرد ، درج می شود.

فیلد اختیاری Option : در این فیلد اختیاری می توان تا حداکثر ۴۰ بایت قرار داد و محتوی اطلاعاتی است که می تواند به مسیریاب ها در یافتن مسیر مناسب کمک کند، یا اطلاعاتی را از آنها طلب می کند .

فیلد Payload: (آخرین فیلد از بسته IP)در این فیلد داده های دریافتی از لایه ی بالاتر قرار می گیرد.

آدرس های IP :

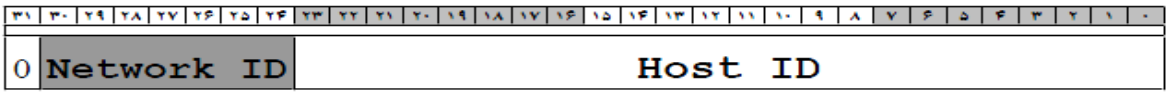
یک آدرس IP را به طور معمول و استاندارد به صورت ۴ عدد دهدهی که با نقطه از هم جدا شده اند می نویسند. بدین ترتیب که این آدرس در محدوده ی ۰.۰.۰.۰ تا ۲۵۵.۲۵۵.۲۵۵.۲۵۵ خواهد بود.

کلاس های آدرس IP : فلسفه کلاسهای آدرس IP بر این اصل استوار است که آدرس ها ، سلسله مراتبی باشند که اساس آدرس های IP ، قالبی به شکل زیر دارد.

آدرس ماشین / آدرس زیرشبکه / آدرس ماشین

آدرس زیر شبکه اختیاری است.

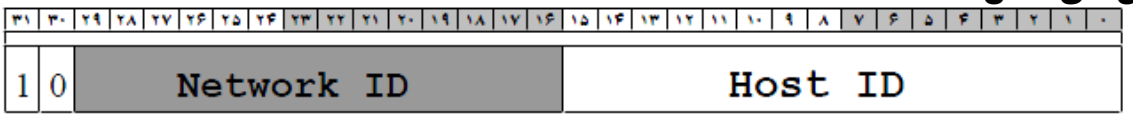
آدرس های کلاس A :



در کلاس A بایت پرارزش در محدوده ی ۰ تا ۱۲۷ تغییر می کند ، مشخصه شبکه دراین کلاس به هیچ وجه نمی تواند ۰ یا ۱۲۷ انتخاب شود ، چرا که این دو عدد در شبکه رزرو شده هستند، بنابراین تعداد شبکه هایی در جهان که می توانند از کلاس A استفاده کنند ، برابر ۱۲۶ است.

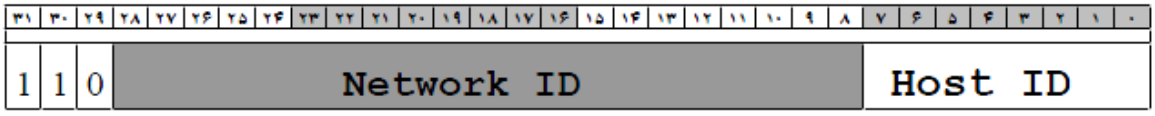
نکته: آدرس ۱۲۷ .۰.۰.۱ در پروتکل IP یک شبکه مشخص نمی کند ، بلکه به صورت قرار دادی به عنوان آدرس حلقه ی بازگشت جهت اهداف اشکال زدایی استفاده شده است چرا که این آدرس عملاً معادل آدرس خود ماشین محلی است ، هر گاه بسته ای را به آدرس ۱۲۷ .۰.۰.۱ بفرستید نرم افزار لایه IP پس از دریافت ، آن را به سوی لایه ی بالاتر بر خواهد گرداند وچنین بسته ای ، هرگز بر روی شبکه ارسال نخواهد شد ، از این آدرس می توان برای آزمایش عملکرد صحیح پشته ی پروتکلی بر روی یک کامپیوتر بهره گرفت ، در ضمن دو برنامه تحت شبکه بر روی یک ایستگاه واحد از طریق این آدرس می توانند داده رد و بدل کنند.

آدرس های کلاس B :



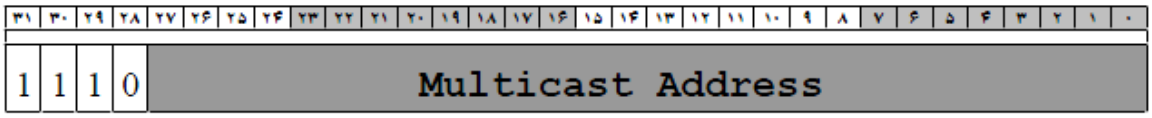
در آدرس های کلاس **B** تعداد ۱۶۳۸۲(۲-۲ ^۱۴) شبکه ی گوناگون قابل تعریف است و هر شبکه می تواند ۶۵۵۳۴ (۲-۲ ^۱۶) ماشین میزبان تعریف کند. اختصاص آدرسهای کلاس **B** برای شبکه های بسیار عظیم مناسب است . هر چند تعداد این شبکه در جهان می تواند تا حدود شانزده هزار عدد باشد ولیکن امروزه عملاً نمی توان آدرس کلاس **B** گرفت ، چرا که تقریباً همه آنها رزرو شده اند. اگر عدد سمت چپ آدرس **IP** بین ۱۲۸ تا ۱۹۱ باشد ، آن آدرس متعلق به کلاس **B** خواهد بود.

آدرس های کلاس C :



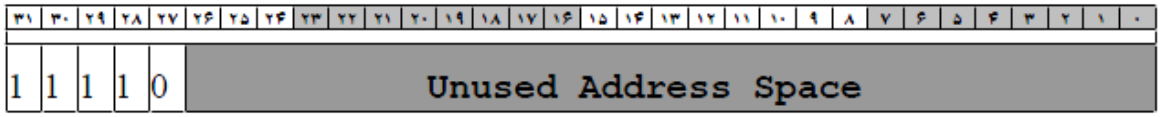
کلاس **C** ، مناسب ترین و پرکاربرد ترین کلاس از آدرس های **IP** است. در این کلاس سه بیت پر ارزش از هر آدرس دارای مقدار۱۱۰ و ۲۱ بیت بعد از سه بیت سمت چپ برای تعیین هویت شبکه مورد نظر بکار می رود. در این کلاس می توان حدود ۲ میلیون شبکه را در جهان آدرس دهی کرد و هر شبکه می تواند تا ۲۵۴ ماشین میزبان تعریف کند. اگرعدد سمت چپ آدرس **IP** بین ۱۹۲ تا ۲۲۳ باشد ، آن آدرس از کلاس **C** خواهد بود.

آدرس کلاس D :



دراین کلاس ۲۸ بیت باقی مانده از کل آدرس برای تعیین آدرس های چند پخشی **Multicast** است. این آدرس ها برای ارسال یک **Datgram** واحد به طور همزمان برای چندین ماشین میزبان کاربرد دارد و به منظور عملیات رسانه ای وچند پخشی مورداستفاده قرار می گیرد.

آدرس کلاس E :



فعلاً این دسته از آدرس ها کاربرد خاصی ندارند و به منظور استفاده در آینده رها شده اند.

آدرس های زیر شبکه:

فرض کنید شرکت شما یک کلاس C ثبت کرده است ، بدین ترتیب شبکه شما توانایی آدرس دهی حداکثر ۲۵۴ ماشین را دارد ، در نظر بگیرید که شما دارای یک شبکه محلی واحد و یکپارچه برای کل شرکت نیستید ، بلکه چند شبکه محلی مجزا برای هر بخش تدارک دیده اید برای آنکه بتوانید زیرشبکه ها را تفکیک کنید باید به گونه ای در بخش شناسه ماشین میزبان (Host ID) زیر شبکه ها نیز مشخص شوند این کار از طریق مفهومی به نام الگوی زیر شبکه (Subnet Mask) انجام می گیرد. مانند ۲۵۵.۲۵۵.۲۵۵.۰ هر گاه یک ماشین بخواهد یک آدرس IP را تحلیل کند الگوی زیر شبکه را با آدرس IP خودش AND می کند (با این کار در حقیقت HostID خودش را صفر می نماید) سپس مجدداً الگو را با آدرس IP مقصد AND می کند و نتیجه ی دو مرحله را با هم مقایسه می نماید ، اگر نتیجه ی دو مرحله یکسان بود هم مشخصه ی شبکه و هم مشخصه ی زیر شبکه در آدرس های مبدأ و مقصد یکی است و هر دو روی یک شبکه محلی قرار دارند.

الگوریتم تنظیم الگوهای زیر شبکه:

الف) با دقت و دوراندیشی کافی تعیین کنید چه تعداد زیر شبکه و چه تعداد ماشین میزبان روی هر زیرشبکه خواهید داشت ، به تعداد زیر شبکه ها و ماشین ها ، عدد ۲ را اضافه کرده و سپس تعیین کنید هر کدام از این اعداد به حداقل چند بیت نیاز دارند.

ب) الگوی ۳۲ بیتی زیرشبکه را به گونه ای تنظیم کنید که در سمت راست آن به تعداد بیتی که برای آدرس دهی ماشین های میزبان نیاز است، صفر قرار بگیرد ، بیت های باقی مانده را هم تماماً برابر ۱ قرار دهید.

ج) الگوی دودویی را به فرم دهدهی نقطه دار تبدیل کنید.

مثال) فرض کنید یک شرکت بزرگ دارای حداکثر ۲۵ شبکه محلی است و هر شبکه ی محلی حداکثر تا ۱۰۰۰ ماشین میزبان را پشتیبانی می کند ، الگوی زیر شبکه برای کلاس B با آدرس۱۳۳.۱۸۹.۰.۰ بدین صورت تنظیم می شود که برای آدرس دهی ۲۵ زیر شبکه محلی ۵ بیت کفایت می کند ، چرا که با ۵ بیت می توان ۳۰ زیرشبکه را تعریف و آدرس دهی کرد. آدرس از کلاس B است پس قسمت ۱۶ بیتی از مشخصه ی ماشین میزبان پس از کسر ۵ بیت (که به عنوان زیر شبکه استفاده خواهد شد) مقدار ۱۱ بیت خواهد بود و حداکثر تعداد ماشین میزبان قابل تعریف برابر۲۰۴۶= (۲-^{11} ۲)است ، الگوی زیرشبکه برای شبکه ی این شرکت به صورت زیر تنظیم می شود:

گاهی اوقات «الگوی زیرشبکه» (Subnet Mask) طبق قاعدهٔ زیر در جلوی آدرس شبکه نوشته می‌شود:

133.189.0.0/21

/21 بدین معناست که ۲۱ بیت **سمت چپ** الگوی زیرشبکه یک و بقیه صفر است. در حقیقت نماد بالا معادل الگوی زیر است:

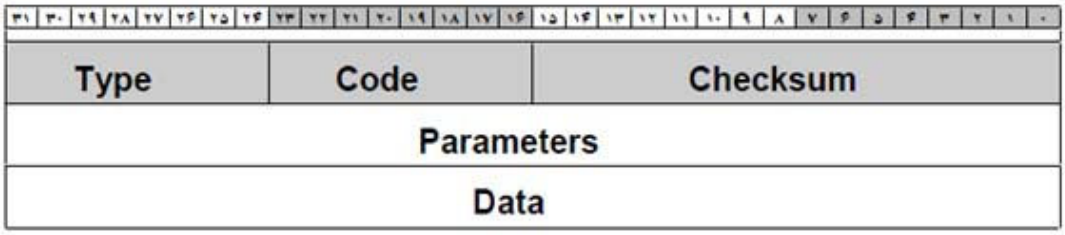
133.189.0.0/255.255.248.0

نماد اول ساده‌تر است و در کتابهای شبکه از آن زیادتر استفاده می‌شود.

پروتکل ICMP (Internet Control Message Protocol):

پروتکل IP پروتکلی بدون اتصال و غیرقابل اعتماد است ، بدون اتصال بدین معناست که مسیریاب هر بسته را بدون هیچگونه هماهنگی با مقصد بسته یا مسیریاب بعدی ارسال می نماید، در ضمن هر مسیر یاب پس از ارسال یک بسته آن را فراموش می کند و منتظر دریافت رسید بسته از گیرنده نخواهد ماند. اگر یک بسته ی IP با خطا به مقصد برسد و یا اصلاً به مقصد نرسد، این پروتکل هیچ اطلاعی در مورد سرنوشت آن به فرستنده ی بسته نمی دهد. پروتکل ICMP در کنار پروتکل IP برای بررسی انواع خطا و ارسال پیام برای مبدأ بسته در هنگام بروز اشکالات ناخواسته استفاده می شود. این پروتکل اشکالات موجود را در قالب یک سری پیام ، گزارش می کند که این پیام خود در یک بسته ی IP قرار می گیرد و از جانب یک مسیریاب یا ماشین مقصد به آدرس فرستنده باز می گردد.

قالب پیام های ICMP :

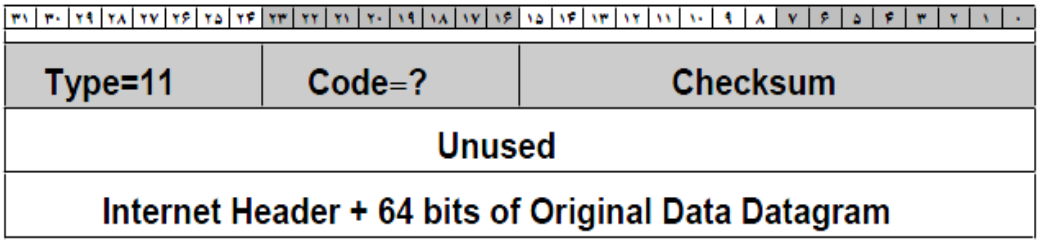


فیلد Type : دراین فیلد عددی قرار می گیرد که بیانگر نوع پیام است و ساختار فیلدهای parmeters و data بسته به عددی که در این فیلد قرار گرفته متفاوت خواهند بود.

فیلد Code : گاهی خود نوع پیام به چند نوع فرعی دیگر تقسیم می شود که کد نوع فرعی در این فیلد قرار می گیرد.

فیلد Checksum : محتوای این فیلد برای سنجش اعتبار و سلامت بسته ICMP مورد استفاده قرار می گیرد. تمام بسته ICMP به صورت دو بایت دو بایت جمع شده و نهایتاً از مکمل ۱ حاصل جمع ، عددی ۱۶ بیتی به دست می آید که درون این فیلد قرار می گیرد.

مثال: پیام(Destination Unreachable) این پیام زمانی صادر می شود که زیرشبکه یا یک مسیریاب نتواند آدرس مقصد را تشخیص بدهد ، یا به هر دلیلی بسته توسط ماشین میزبان تحویل گرفته نشود. ساختار بسته ی حامل این پیام به صورت زیر است:



معنای شماره های مختلف در فیلد کد به شرح زیر است:

۰ : شبکه مورد نظر در دسترس نمی باشد.

۱ : ماشین میزبان مورد نظر در دسترس نمی باشد.

۲ : پروتکل مورد نظر تعریف نشده است.

۴ : شماره ی پورت مورد نظر وجود ندارد.

۵ : اندازه ی بسته بزرگ است و نیاز به شکستن دارد ، درحالی که اجازه شکستن داده نشده است.

نکته: در تمام بسته های **ICMP** که به عنوان گزارش خطا برای ماشین مولد بسته ی **IP** فرستاده می شود ، سرآیند بسته ی حذف شده به همراه ۶۴ بیت اول از داده (که خود ۸ بایت اول از سرآیند بسته ی لایه ی انتقال است) برای مولد ارسال می شود تا در صورت نیاز به بررسی عامل خطا ، اطلاعاتی از بسته حذف شده در اختیارش باشد.

پروتکلARP(Address Resolution Protocol):

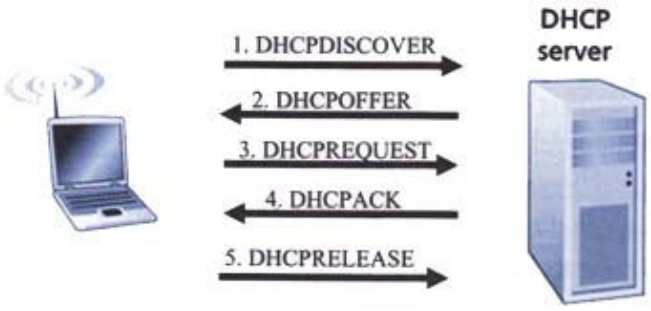
آدرس های **IP** فقط در لایه ی شبکه قابل شناسایی و تحلیل هستند، یک بسته **IP** قبل از ارسال برروی کانال از لایه ی واسط شبکه عبور می کند و ضمن اضافه شدن اطلاعات لازم و تشکیل یک فریم روی کانال فیزیکی ارسال می شود. لایه ی واسط شبکه برای ارسال یک فریم نیاز به آدرس فیزیکی ماشین مقصد (**MAC**) دارد ، بنابراین وقتی پروتکل **IP** می خواهد یک بسته اطلاعاتی را روی شبکه بفرستد باید به نحوی آدرس فیزیکی اولین ماشینی که بایستی با آن ارتباط برقرار کند را بداند ، این ماشین می تواند مسیریاب پیش فرض او باشد یا می تواند آدرس فیزیکی مقصد روی همین شبکه ی محلی باشد. حال فرض کنید ایستگاهی آدرس **IP** ماشینی را که می خواهد با آن ارتباط برقرار کند را می داند ولی آدرس فیزیکی او را نمی داند ، چه کاری می تواند انجام بدهد؟ ، باید از پروتکل **ARP** بهره ببرد. وظیفه پروتکل **ARP** این است که یک بسته ی فراگیر روی یک شبکه محلی منتشر کند که این بسته در حقیقت آدرس فیزیکی یک آدرس **IP** مشخص را سوال می کند ، با توجه به آنکه بسته های فراگیر توسط تمام ماشین های روی شبکه محلی دریافت می شود ، ماشینی که آدرس **IP** خودش را درون این بسته می بیند ، فوراً به آن پاسخ می دهد و آدرس فیزیکی خود را برای ارسال کننده آن بسته پس می فرستد. بر خلاف پروتکل **ICMP** که روی پروتکلIP قرار می گیرد،پروتکل **ARP** مستقیماً بر روی پروتکل لایه ی فیزیکی عمل می کند ، یعنی یک بسته ی **ARP** ساخته شده و درون فیلد داده از فریم لایه ی فیزیکی قرار گرفته ، روی کانال ارسال می شود. مسئله دیگری که ممکن است در هنگام بکارگیری پروتکلARP رخ بدهد آن است که ، ممکن است یک دستگاه روی شبکه محلی دیگری باشد و بنابراین پاسخی نمی رسد ، درچنین حالتی دو راه حل وجود دارد :

الف)وقتی مسیریابی که به آن شبکه متصل است می بیند ،آدرس مقصد که توسطARP سوال شده، روی یک شبکه محلی دیگر واقع است ، در پاسخ آن آدرس فیزیکی خودش را به ایستگاه سوال کننده ارسال می دارد به این روش Proxy ARP گفته می شود.

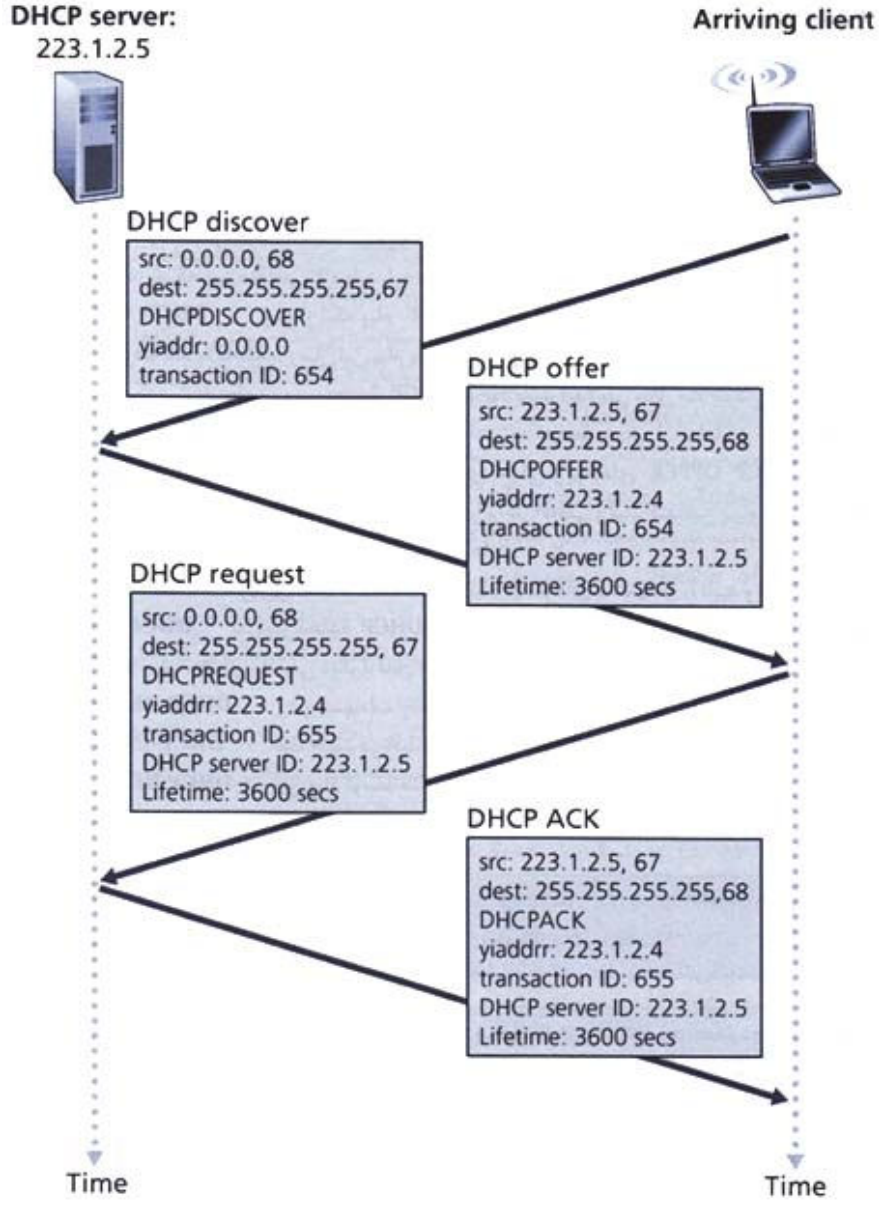
ب)ایستگاه ها خودشان موظفند به روشی که در مبحث الگوی زیر شبکه اشاره شد مستقلاً محلی یا خارجی بودن ماشین مقصد را تشخیص داده و در صورت خارجی بودن آدرس فیزیکی مسیر یاب پیش فرض را انتخاب نمایند.

پروتکل DHCP (Dynamic Host Configuration Protocol) :

پروتکلDHCP این امکان را فراهم آورده که بتوان آدرس IP ایستگاه ها را هم به صورت دستی و هم به صورت خودکار به آن ها انتساب داد. پروتکل DHCP متکی به یک سرویس دهنده ی ویژه درشبکه است تا به ماشین هایی که تقاضای آدرسIP می کنند ، آدرس و اطلاعات لازم را تقدیم نماید. لازم نیست که این سرویس دهنده ، بر روی همان LAN باشد که ماشین متقاضی آدرس برروی آن واقع شده است ، و از آنجایی که ممکن است دسترسی به این سرویس دهنده از طریق پخش فراگیر بسته های تقاضا میسر نباشد، بنابراین بر روی هر LAN به یک عامل رله ی DHCP(DHCP Relay Agent) نیاز است.
ماشینی که تازه بوت شده برای بدست آوردن آدرس IP خود، بسته ای بنام DHCP DISCOVER را بصورت فراگیر بر روی LAN خود منتشر می کند، عامل رله یDHCP در هر LAN تمام بسته های پخش شده ی DHCP را می گیرد ، و وقتی متوجه شود که یک بسته از نوع DHCP DISCOVER است ، آن را به صورت تک پخشی برای سرویس دهنده ی اصلی DHCP که می تواند بر روی شبکه ای دور واقع باشد می فرستد. عامل رله یDHCP فقط به آدرس IP از سرویس دهنده DHCP احتیاج دارد و توسط مسئول شبکه نصب و پیکربندی می شود. در پاسخ به پیام DHCP DISCOVER ، سرویس دهنده یا سرویس دهنده های DHCP خود را با پیام DHCP OFFER معرفی کرده و پارامتر های مورد نیاز را پیشنهاد می دهند. گیرنده ی تازه وارد از بین پیشنهادهای رسیده ، فقط یکی را بر گزیده و در پیام بعدی برای او مستقیماً یک پیام DHCP REQUEST می فرستد و از او می خواهد که پارامتر های پیشنهاد شده را برای او قطعی و ثبت کند. سرویس دهنده یDHCP در پاسخ با ارسال پیام DHCP ACK ،پارامترهای لازم پیکر بندی ماشین را برای او فرستاده و آدرسIP او را ثبت می کند ، پس از تأیید این پارامترها توسط سرویس دهنده ، ماشین تازه وارد می تواند ، خود را با این پارامترها پیکر بندی کرده و کارش را آغاز کند، سر انجام هر گاه این ماشین بخواهد شبکه راترک کند، بایستی با پیام DHCP RELEASE تقاضای آزاد شدن آدرسIP خود را اعلام کرده و ازشبکه خارج شود. البته چون آدرس اجاره ای باید تمدید شود لذا اگر مدت حضور ماشین در شبکه زیاد باشد در این بین باید تعداد بسته یDHCP REQUEST برای تمدید مهلت اجاره به سوی سرویس دهنده ارسال گردد ، و در غیر این صورت پس از مهلت مقرر آدرس نامعتبرخواهد بود.



فرآیند مذاکرهٔ بین مشتری و سرویس‌دهندهٔ DHCP



بخش ۴

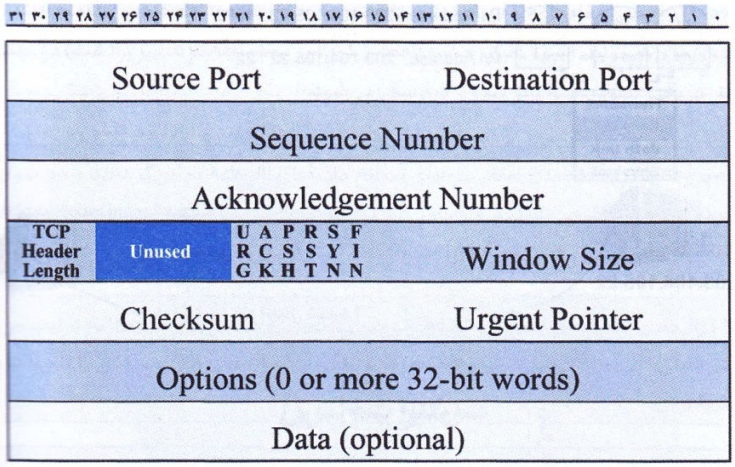
لایه ی انتقال در شبکه ی اینترنت

وظیفه لایه انتقال : بهره گیری از خدمات لایه ی IP که سریع ، ساده و در عین حال غیر مطمئن و ناکارآمد است و ارائه خدماتی مطمئن ، ساختار یافته و شفاف به برنامه های کاربردی در لایه ی بالاتر که برنامه نویس از درگیری با جزئیات زیرساخت شبکه و مشکلات انتقال و مسایلی از این قبیل به دور باشد.

در لایه انتقال دو پروتکل به نام های : TCP(Transmission Control Protocol) وUDP (User Datagram Protocol) تعریف شده اند.

تعریف شماره پورت : هر پروسه برای تقاضای برقراری یک ارتباط با پروسه ای دیگر روی شبکه یک شماره شناسایی برای خود بر می گزیند ، به این شماره شناسایی ، شماره ی پورت گفته می شود . بنابراین زوج آدرس IP و شماره ی پورت می تواند یک پروسه یکتا و واحد را بر روی هر ماشین در دنیا مشخص نماید.

ساختار بسته های پرتکل TCP :



فیلد Source Port : در این فیلد یک شماره ۱۶ بیتی به عنوان شماره ی پورت پروسه ی مبدأ که این بسته را جهت ارسال تولید کرده، قرار خواهد گرفت.

فیلد Destination Port : در این فیلد شماره ی پورت پروسه ی مقصد که آن را تحویل خواهد گرفت، تعیین خواهد شد.

نکته: برخی از پروسه های کاربردی و استاندارد دارای شماره ی پورت استاندارد و جهانی هستند. مثلاً سرویس دهنده ی پست الکترونیکی دارای شماره ی پورت ۲۵ است.

فیلد Sequence Number : این فیلد ۳۲ بیتی شماره ی ترتیب آخرین بایتی را که در فیلد داده از بسته ی جاری قرار دارد نشان می دهد. دقت شود که شماره ی ترتیب اولین بایت از ۰ شروع نمی شود بلکه ازیک عدد تصادفی که در هنگام برقراری ارتباط ، به اطلاع طرفین می رسد ، شروع خواهد شد.

فیلد Acknowledgement Number : این فیلد۳۲ بیتی ، شماره ترتیب بایتی را که فرستنده ی بسته منتظر دریافت آن است ، تعیین می کند .

فیلد TCP Header Length : عددی که در این فیلد ۴ بیتی قرار می گیرد طول سرآیند بسته **TCP** را بر مبنای کلمات ۳۲ بیتی تعیین میکند. به عنوان مثال اگر در این فیلد عدد ۷ قرار بگیرد طول سرآیند مقدار ۲۸=۷*۴ بایت خواهد بود از آنجا که در فیلد اختیاری **Option** می تواند اطلاعاتی با اندازه متغیر قرار بگیرد بنابراین گیرنده یک بسته **TCP** باید بتواند مرز بین سرآیند بسته و قسمت داده را تشخیص بدهد بنابراین عددی که در این فیلد قرار می گیرد می تواند به عنوان یک اشاره گر محل شروع داده ها را در یک بسته **TCP** تعیین کند.

بیت های FLAG :

۱- بیت **URG** : درصورتی که این بیت مقدار یک داشته باشد معین می کند که در فیلد **Urgent Pointer** مقداری قابل استناد و معتبر قرار دارد و بایستی مورد پردازش قرار گیرد.
۲- بیت **ACK** : اگر در این بیت مقدار یک قرار گرفته باشد ، نشان می دهد که عددی که در فیلد **Acknowledgement Number** قرار گرفته است ، دارای مقداری معتبر و قابل استناد است.
۳- بیت **PSH** : اگر در بیت**PSH (Push)** ، مقدار یک قرار گرفته باشد ، نشان می دهد که فرستنده ی اطلاعات ، از گیرنده تقاضا می کند که داده های موجود در این بسته را بافر نکند و در اسرع وقت آن راجهت پردازش های بعدی تحویل برنامه ی کاربردی صاحب آن بدهد.
۴- بیت **RST** : اگر در این بیت مقدار یک قرار بگیرد، ارتباط بصورت یکطرفه و ناتمام قطع خواهد شد.
۵- بیت **SYN** : این بیت نقش اساسی در بر قراری یک ارتباط(اتصال یا همان **Connection**) ، بازی می کند.
۶- بیت **FIN** : اگر یکی از طرفین ارتباط ،داده ی دیگری برای ارسال نداشته باشد،در هنگام ارسال آخرین بسته ی خود ، این بیت را یک می کند و در حقیقت ارسال اطلاعات خودش را یکطرفه قطع می کند.

فیلد Windows Size : مقدار قرار گرفته در این فیلد مشخص می کند که فضای بافر گیرنده چند بایت دیگر ظرفیت خالی دارد، یعنی به طرف مقابل اعلام می کند که مجاز است از بایت با شماره ی ترتیبی که در فیلد **Acknowledgement** مشخص شده است ، حد اکثر به اندازه ی مقداری که در این فیلد درج شده ، ارسال داشته باشد و در غیر اینصورت فضای کافی برای دریافت داده ها وجود نداشته و ناگزیر دور ریخته خواهند شد. اگر مقدار این فیلد صفر باشد بدین معناست که بافر گیرنده تماماً پر شده است و امکان دریافت داده های بعدی وجود ندارد و پروسه ی فرستنده متوقف خواهد شد.

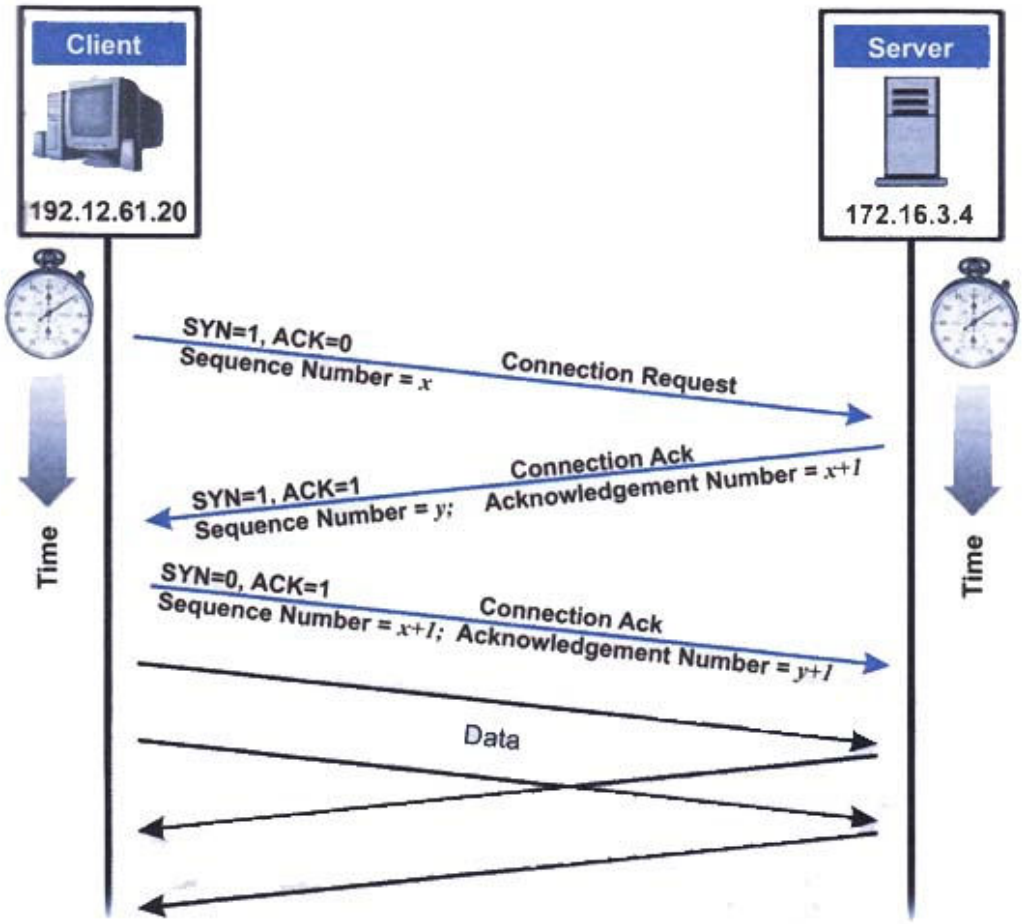
فیلد Checksum : در این فیلد۱۶ بیتی ،کد کشف خطا قرار می گیرد.

فیلد Urgent Pointer : در این فیلد یک عدد به عنوان اشاره گر قرار می گیرد که موقعیت داده های اضطراری را درون بسته **TCP** معین می کند. تولید این داده ها زمانی اتفاق می افتد و ارسال می شود که عملی شبیه وقوع وقفه ها در هنگام اجرای یک برنامه کاربردی رخ بدهد. بدون آنکه ارتباط قطع شود ، داده های لازم در همین بسته ی جاری ارسال خواهند شد، دقت کنید که داده های اضطراری توسط برنامه ی کاربردی در لایه ی بالاتر پردازش خواهند شد و برای پروتکل**TCP** کار بردی ندارد.

فیلد Options : این فیلد اختیاری است و در آن مقادیری نظیر حداکثر طول بسته یTCP قابل دریافت ، یا توافق بر روی مکانیزم هایی متفاوت با قراردادهای پیش فرض قرار می گیرد. همچنین در بسیاری از مواقع برای آنکه طول بسته ضریبی از ۴ باقی بماند این فیلد با کدها ی بی ارزش پر می شود.

روش برقراری ارتباط در پروتکلTCP:

برای برقراری ارتباط در پروتکلTCP،از روش دست تکانی سه مرحله ای استفاده می شود.



برای خاتمه ارتباط ، طرفی که داده هایش برای ارسال تمام شده است ، یک بسته ی TCP ارسال می نماید که در سرآیند آن بیتFIN را یک قرارداده است. طرف مقابل، این درخواست را دریافت و با ختم یکطرفه ی ارتباط آن موافقت می کند ولی چون ارتباط به صورت یک طرفه ختم می شود ، طرف مقابل می تواند تا جایی که داده دارد آن ها را ارسال کند و نهایتاً در آخرین بسته بیت FIN را یک بگذارد تا پس از تصدیق آن ارتباط به صورت دو طرفه ختم شود.

زمان سنج ها در پروتکلTCP:

۱- **Retransmission Timer :** پس از برقراری یک ارتباط ، وقتی بسته ای برای پروسه ی مقصد ارسال می شود ، ضمن نگهداری موقت آن در یک بافر برای آن یک زمان سنج تنظیم و فعال می شود و اگر در مهلت مقرر، پیغام دریافت آن نرسید ، آن بسته از نو برای مقصد ارسال خواهد شد. این زمان سنج اختصاراً **RT** نامیده می شود. سوالی که در اینجا مطرح می شود این است که مقدار پیش فرض این زمان سنج چقدر باید باشد؟ در شبکه ها ی محلی سریع ، زمان رفت یک بسته و برگشت پیغام دریافت آن در حدود کسری از میلی ثانیه خواهد بود در حالی که در شبکه **WAN** ، این زمان رفت و برگشت می تواند تا چندین ثانیه طول بکشد، بهترین راه تنظیم این زمان سنج استفاده از الگوریتم های پویا است.

۲- **Keep-Alive Timer :** ممکن است طرفین یک ارتباط به هر دلیلی ارسال اطلاعات را موقتاً متوقف کنند و هیچ داده ای مبادله نشود هر چند ارتباط**TCP** فعال و باز باشد ، از سوی دیگر ممکن است یکی از طرفین به دلیلی مثل خرابی سخت افزاری یا نرم افزاری بدون اطلاع ارتباط را رها کرده باشد، برای تمایز بین این دو حالت فرستنده اطلاعات با استفاده از این زمان سنج در بازه های زمانی منظم یک بسته ی **TCP** که خالی از هر گونه داده ای است برای مقصد ارسال می کند و در صورتی که پیغام دریافت آن باز گردد نشان دهنده ی آن است که **TCP** فعال و باز است ، در غیر اینصورت ارتباط **TCP** به صورت یکطرفه قطع شده و تمام بافرها و ساختمان داده ی **TCP** آزاد می شوند. زمان پیش فرض این زمان سنج ۳۰ تا ۴۵ ثانیه است.

۳- **Persistence Timer :** در پروتکل **TCP** ، وقتی یکی از طرفین ارتباط ، مقدار فضای بافر آزاد خود را در فیلد **WindowSize** ، صفر اعلام کند ناگزیر پروسه طرف مقابل متوقف خواهد شد. در چنین حالتی پس از آنکه ، مقداری از فضای بافر پر شده تخلیه شد ، این موضوع باید به طرف مقابل گزارش شود تا سیستم عامل پروسه ی بلوکه شده را احیاء کرده و ادامه ارسال از طرف مقابل ممکن باشد. در غیر این صورت بن بست و تأخیری بی نهایت برای پروسه به وجود خواهد آمد. با استفاده از این زمان سنج پس از آزاد شدن فضای بافر در فواصل زمانی منظم یک بسته ی **TCP** برای پروسه ی بلوکه شده ارسال می شود تا ضمن آگاهی از آخرین وضعیت فضای بافر در صورت آزاد شدن فضا در پنجره ی گیرنده ، پروسه بتواند احیاء شود.

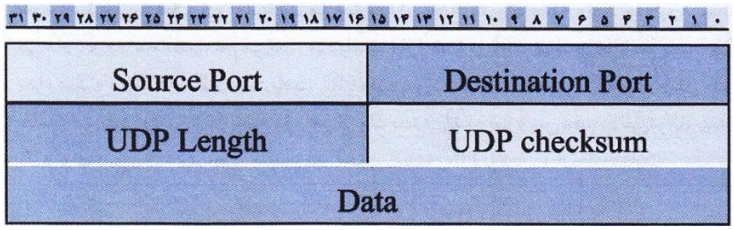
۴- **Quiet Timer** : ممکن است یک ارتباط **TCP** بسته شود ولی هنوز بسته هایی سرگردان بر روی شبکه وجود داشته باشند و پس از بسته شدن ارتباط **TCP** به مقصد برسند، لذا در این پروتکل پس از بسته شدن یک ارتباط با شماره ی پورت خاص بقیه ی پروسه ها تا مدتی حق استفاده از شماره ی پورتی که اخیراً بسته شده را ندارند. این زمان را این تایمر مشخص می نماید ، مقدار پیش فرض این زمان سنج دقیقاً دو برابر مقدار زمان پیش فرض زمان حیات بسته **IP** بر حسب ثانیه است(بین ۳۰ تا ۱۲۰ ثانیه).

۵- **Idle Timer :** این زمان سنج برای آن است که اگر تلاش برای تکرار ارسال یک بسته ، بیش از حد متعارف انجام شود ، ارتباط **TCP** را به صورت یکطرفه رها کرده و قطع نماید. مقدار معمول این زمان سنج بین ۱۲۰ تا ۳۶۰ ثانیه است.

پروتکل UDP:

پروتکل TCP ، پروتکلی اتصال گرا است و لزوم برقراری یک اتصال قبل از هرگونه مبادله داده، می تواند بین چندین میلی ثانیه تا چندین ثانیه طول بکشد ، در ضمن معطلی برای بازگشت پیغام های ACK ، یک پروسه کاربردی را با تأخیر مواجه خواهد کرد، برای برخی از کاربرد ها مانند ارسال صدا و تصویر این زمان های تلفاتی قابل تحمل نیست و سرعت در رسیدن یک بسته به مقصد ضروری تر است. در لایه انتقال از مدلTCP/IP برای چنین کاربرد هایی یک پروتکل ساده و سریع به نامUDP معرفی شده که ذاتاً بدون اتصال (Connectionless) است، یعنی بدون آگاهی از سرنوشتی که یک بسته خواهد داشت و بدون هماهنگی با هیچ موجودیتی در شبکه آن را به سمت مقصد ارسال می کند.

ساختار بسته یUDP :



فیلد Source Port : دراین فیلد یک شماره ی ۱۶ بیتی، به عنوان شماره ی پورت پروسه ی مبدأ که این بسته را تولید کرده قرار خواهد گرفت.

فیلد Destination Port : در این فیلد شماره ی پورت پروسه ی مقصد که آن را تحویل خواهد گرفت، تعیین خواهد شد.

فیلد UDP Length :در این فیلد طول بسته یUDP ، بر حسب بایت (شامل سر آیند داده ها) درج می شود.

فیلد UDP Checksum : در این فیلد۱۶ بیتی که کشف خطا درج می شود ، به کار گیری این فیلد اختیاری است و در صورت عدم نیاز به آن تمام بیت های آن به صفر تنظیم می شود.

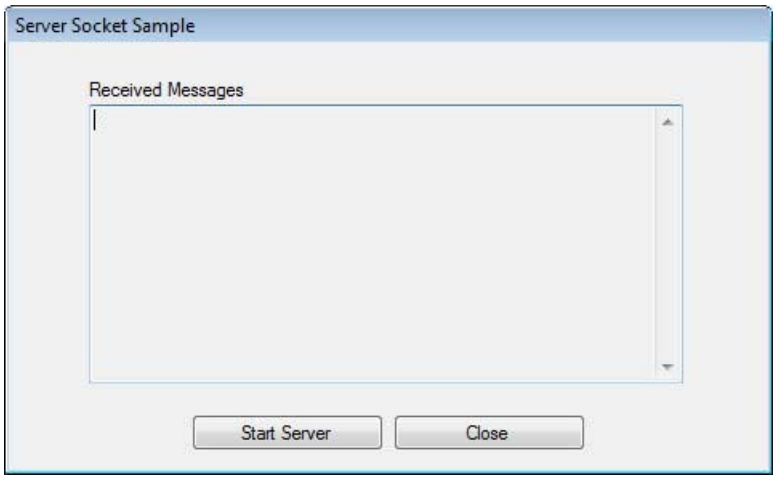
ماشین های LittleEndian و BigEndian : ساختار بسته های IP وTCP در قالب کلمات۳۲ بیتی سازماندهی می شود، ماشین های متفاوتی که در دنیا وجود دارند ، کلمات دو بایتی چهار بایتی را به روشی یکسان درون حافظه ذخیره نمی کنند. ماشینهایی که ابتدا بایت کم ارزش وسپس بایت پر ارزش را ذخیره می کنند، ماشین هایLittleEndian نامیده می شوند، کامپیوترهای شخصی با پردازنده های سری 80X86 و پنتیوم از این دسته هستند ، ماشین هایی که ابتدا بایت پرارزش و سپس بایت کم ارزش را ذخیره می کنند ماشین هایBig Endian نامیده می شوند. کامپیوترهای سری SUN از این دسته هستند، پروتکل TCP/IP استاندارد ماشین هایBig Endian را مبنا قرار داده است، لذا در تمام ماشین هایLittle Endian قبل از ارسال بسته های IP باید فیلدهای دو بایتی و چهار بایتی درون حافظه به گونه ای تنظیم و مقداردهی شوند تا در هنگام ارسال بسته روی خط ، ابتدا بایت پر ارزش ارسال شود.

بخش پنجم

برنامه نویسی سوکت به کمک C#

برنامه اول :

برنامه ی زیر شامل یک سرور و یک کلاینت است . سرور یک سوکت تعریف می کند و به آن گوش می دهد تا زمانی که کلاینت درخواست ارتباط با آن سوکت را بدهد، سپس سرور ارتباط را تایید کرده و آماده دریافت پیام ها از کلاینت می شود.



شکل بالا نمایی از سرور برنامه است

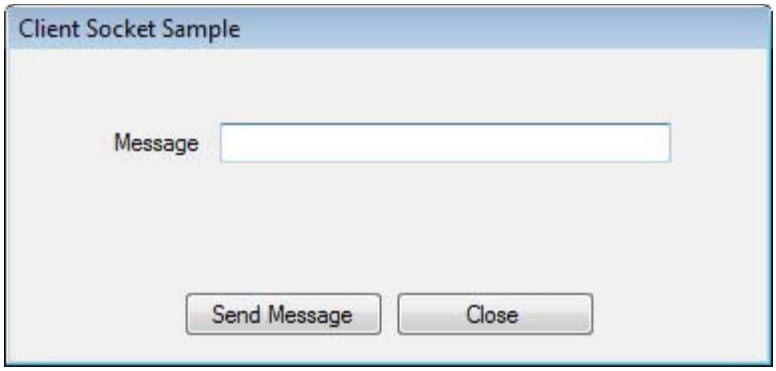
در برنامه ی سرور کدهای زیر نوشته می شود :

```
// کتابخانه ی توابعی که در این برنامه به آنها نیاز داریم
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Net;
using System.Net.Sockets;
// ***************************************************
namespace ServerSocketSample
{
    public partial class Form1 : Form
    {
        // تعاریف اولیه برنامه
        // یک شیٔ سوکت برای سوکت سرور
        private Socket listener;
        // یک شیٔ سوکت برای کلاینت
        private Socket client;
        // رشته ای برای آدرس آی پی ، در آن مقداری را قرار داده ایم که به خود سیستم اشاره می کند
        private string ClientIPAddress = "127.0.0.1";
        // رشته ای برای آدرس پورت ، مقداری که در آن قرار داده شده در محدوده ی پورتهای رزرو شده نیست
        private int ClientPortNo = 10000;
        // رشته ای جهت تعیین حداکثر کانکشن فعال
        private int MaxConnections = 5;
        // یک متغیر عددی جهت تعیین حداکثر طول مجاز یک پیام
        private int MessageLength = 200;
        // آرایه ای از بایتها جهت دریافت پیامها
        private byte[] ReceivedBuffer;
        // ***************************************************
```

```
        public Form1()
        {
            InitializeComponent();
        }
        // ******************************************************
        // قطعه کد مربوط به دکمه بستن برنامه
        private void btnClose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        // ******************************************************
        // قطعه کد مربوط به دکمه استارت
        private void btnStartServer_Click(object sender, EventArgs e)
        {
            SetButtonEnabled(btnStartServer, false);
            SetButtonEnabled(btnClose, false);
            // توسط دو خط بالا دکمه های استارت و بستن برنامه را غیر فعال می کنیم
            // برای این کار یک تابع مخصوص تعریف نموده ایم و از آن استفاده میکنیم
            listener = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
            // به سوکت حافظه اختصاص می دهد new
            // بدین معنا است که شبکه ای که از آن استفاده میکنیم دارای آی پی نسخه 4 است AddressFamily.InterNetwork
            //  برای سوکت هایی است که می خواهند داده ها را بصورت Stream تبادل کنند SocketType.Stream
            //  نوع پرتکل ما را مشخص می کند ProtocolType.Tcp
            // اکنون مي بايد آدرس IP و يك Port به سوكت مان اختصاص دهيم
            EndPoint ipEndPoint = new IPEndPoint(IPAddress.Parse(ClientIPAddress), ClientPortNo);
            listener.Bind(ipEndPoint);
            // كلاس IPEndPoint براي مشخص نمودن يك نود يا يك كامپيوتر در شبكه به كار مي رود
            listener.Listen(MaxConnections);
            // عدد موجود در MaxConnections نشان می دهد که حداکثر چند ارتباط مي تواند در صف قرار گيرد
            // حال مي بايد تقاضاي كانكت شدن كلاينت را بپذيريم
            client = listener.Accept();
            // در خط بالا برنامه متوقف می ماند تا یک کلاینت درخواست کانکت شدن بدهد
            ReceivedBuffer = new byte[MessageLength];
            // به بافر به طول استاندارد پیام ها حافظه اختصاص دادیم
            // اکنون یک Delegate تعریف می کنیم و تابع OnReceivedData را به آن نسبت می دهیم
            AsyncCallback callback = new AsyncCallback(OnReceivedData);
            client.BeginReceive(ReceivedBuffer, 0, MessageLength, SocketFlags.None, callback, null);
        }
        // ******************************************************
        // تابع زیر جهت دریافت اطلاعات از سوکتی است که ارتباط آن برقرار شده
        private void OnReceivedData(IAsyncResult ar)
        {
            int nBytesRec = client.EndReceive(ar);
            if (nBytesRec <= 0)
            {
                // در صورتی که مقدار بازگشتی از سوکت کوچکتر مساوی صفر باشد ارتباط قطع شده
                // پس سوکت ها را می بندیم ، دکمه ها را فعال می کنیم و از تابع دریافت اطلاعات خارج می شویم
                client.Close();
                listener.Close();
                SetButtonEnabled(btnStartServer, true);
                SetButtonEnabled(btnClose, true);
                return;
            }
            // در صورتی که ارتباط برقرار باشد و مقدار بازگشتی در بافر داشته باشیم اعمال زیر را انجام می دهیم
            string message = Encoding.ASCII.GetString(ReceivedBuffer);
            // مقدار موجود در بافر را از حالت دنباله ای از بایت ها به صورت رشته در می آوریم و
            // در یک متغیر ذخیره می کنیم
            message = message.Trim();// کاراکتر های اسپیس اضافی در انتهای رشته را حذف می کنیم
            // توسط تابع زیر متن پیام را در تکست باکس فرم نمایش می دهیم
            ShowMessage(message);
            // مجددا درون خود تابع یک Delegate تعریف می کنیم و خود تابع را به آن نسبت میدهیم
            AsyncCallback callback = new AsyncCallback(OnReceivedData);
            client.BeginReceive(ReceivedBuffer, 0, MessageLength, SocketFlags.None, callback, null);
        }
```

```
        // تابع زیر یک دکمه و یک پارامتر منطقی را به عنوان ورودی دریافت کرده و
        // دکمه ی مورد نظر را با توجه به مقدار منطقی ، فعال یا غیر فعال می کند
        private void SetButtonEnabled(Button btn, bool enabled)
        {
            System.Threading.ThreadStart todo = delegate
            {
                btn.Enabled = enabled;
            };
            btn.Invoke(todo);
        }
        // ******************************************************
        // تابع زیر  رشته ای را که از پارامتر ورودی خود دریافت کرده در
        // تکست باکس فرم نمایش می دهد
        // برای این کار باید دستورات مورد نظر را در یک نخ جدید از پردازش ها تعریف کنیم
        private void ShowMessage(string message)
        {
            System.Threading.ThreadStart todo = delegate
            {
                txtReceivedMessages.Text += message + "\r\n";
            };
            txtReceivedMessages.Invoke(todo);
        }
        // ******************************************************
        private void Form1_Load(object sender, EventArgs e)
        {

        }
    }
}
```

برنامه ی کلاینت :



شکل بالا نمایی از کلاینت برنامه است

در برنامه ی کلاینت کد های زیر نوشته می شوند :

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Net;
using System.Net.Sockets;
```

```
namespace ClientSocketSample
{
    public partial class Form1 : Form
    {
        private Socket server;
        private string ServerIPAddress = "127.0.0.1";
        private int ServerPortNo = 10000;
        private int MessageLength = 200;
        //*****************************************************
        public Form1()
        {
            InitializeComponent();
        }
        //*****************************************************
        private void Form1_Load(object sender, EventArgs e)
        {
            try // از این دستور برای کنترل خطا های زمان اجرا استفاده می شود
            {   // در صورتی که یکی از دستورات داخل بلاک در حین اجرا دچار خطا شود
                // بجای متوقف شدن برنامه کنترل اجرای برنامه به قسمت catch منتقل می شود
                server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
                IPAddress ipAddress = IPAddress.Parse(ServerIPAddress);
                // برای سوکت تعریف شده حافظه اختصاص دادیم و آدرس و پورت سرور را در آن ست نمودیم
                server.Connect(ipAddress, ServerPortNo);
                // به سوکت تعریف شده درخواست ارتباط دادیم
                // در صورتی که سرور فعال باشد و پورت و آدرس آی پی صحیح باشند ارتباط برقرار می شود
                // در غیر این صورت برنامه دچار خطا شده و به قسمت کنترل خطا در Catch میرود
            }
            catch (SocketException)
            {
                // در صورت بروز خطا در اتصال به سرور پیغام زیر صادر شده و برنامه بسته می شود
                MessageBox.Show("Cannot connect to Server");
                this.Close();
            }
        }
        //*****************************************************
        // Close کدهای مربوط به دکمه
        private void btnClose_Click(object sender, EventArgs e)
        {
            server.Close();
            this.Close();
        }
        //*****************************************************
        //   Send Message کد های مربوط به دکمه
        private void btnSend_Click(object sender, EventArgs e)
        {
            byte[] buffer = PadMessage(txtMessage.Text);
            // مقدار موجود در تکست باکس را به تابع PadMessage داده و خروجی آن را
            // که دنباله ای از بایت ها است را در متغیر buffer ذخیره می کند
            server.Send(buffer, 0, MessageLength, SocketFlags.None);
            // توسط دستور بالا مقدار موجود در buffer را بر روی کانکشنی که ایجاد کردیم
            // برای سرور ارسال می کنیم
            txtMessage.Text = "";// تکس باکس را خالی می کنیم
        }
        //*****************************************************
        // تابع زیر یک رشته از پارامتر ورودی دریافت کرده
        // در صورتی که طول آن از طول مجاز پیام کم تر بود
        //آنقدر به آن کاراکتر فاصله اضافه می کند تا به طول مجاز برسد
        // سپس آن را تبدیل به دنباله ای از بایت ها کرده و به عنوان
        // مقدار بازگشتی خود باز می گرداند
        private byte[] PadMessage(string message)
        {
            while (message.Length < MessageLength)
                message += " ";
            byte[] buffer = Encoding.ASCII.GetBytes(message);
            return buffer;
        }
    }
}
```

تمرین اول : برنامه قبل را طوری تغییر دهید که در سرور یک دکمه **Stop Server** اضافه شود ، به طوری که هر زمان سرور استارت شد این دکمه فعال گردد و اگر دکمه **Stop Server** کلیک شد سرور غیر فعال شده ، این دکمه غیر فعال گردد و دکمه ی استارت فعال شود . همچنین کلاینت را طوری تغییر دهید که بجای آنکه در روال **Form Load** برنامه به سرور کانکت شود ، هر زمان اقدام به ارسال پیام نمودیم ، اگر کانکت نبود ابتدا کانکشن خود را برقرار نموده و سپس پیغام را ارسال نماید و در صورت بروز مشکل در ارتباط پیغام مناسب نمایش دهد .

نکته : در برنامه ی سرور بعلت استفاده از دستور **Listiner.Accept();** که سنکرون می باشد ، برنامه تا وقتی کلاینت به سرور درخواست اتصال بدهد قفل می شود ، برای رفع این مشکل می توانید از **Socket.BeginAccept Method** که آسنکرون می باشد ، و یا از **Threading** استفاده نمایید .

تمرین دوم : تمرین اول را به یک تکست چت دو طرفه تبدیل نمایید . برنامه قابلیت آن را داشته باشد که بر روی یک شبکه با گرفتن آدرس **IP** سیستم **(Local Connection)** کار کند.

نکته : برای بدست آوردن آدرس **IP** سیستم می توانید در **Command Prompt** سیستم از دستور **ipconfig** استفاده نمایید.

تمرین سوم : برنامه ای بنویسید که در آن **N** کلاینت به یک سرور کانکت شوند ، بدین صورت که هر کلاینت به محض اجرای برنامه یک نام و **IP** سرور را از کاربر بگیرد و خود را به سرور کانکت نماید . هر کلاینتی که به سرور کانکت شد نامش در لیست کلاینت های سرور قرار بگیرد و هرگاه ارتباط کلاینت قطع شد یا برنامه را بست ، نامش از لیست کلاینت های سرور حذف شود . کلاینت ها بتوانند برای هم پیام ارسال کنند ، بدین صورت که وقتی کلاینتی پیامی ارسال نمود ، سرور پیام او را دریافت نموده و برای تمام کلاینت ها به غیر از فرستنده ی پیام ارسال نماید.

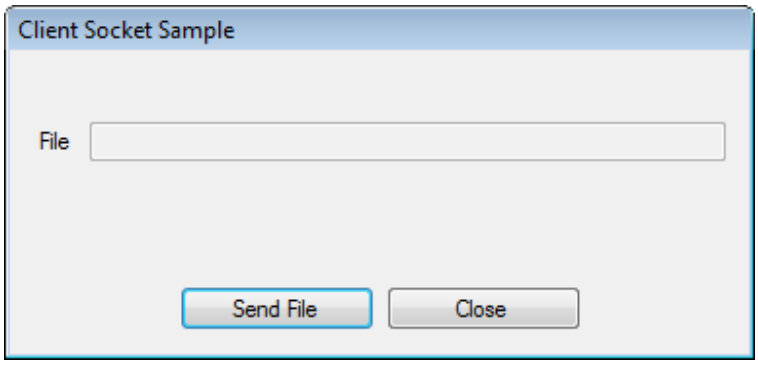
برنامه دوم :

در این برنامه یک فایل متنی کم حجم از کلاینت به سرور انتقال داده می شود.

برای این منظور اعمال زیر را در برنامه انجام خواهیم داد :

۱- فایل متنی مورد نظر را توسط یک شیء **OpenFileDialog** انتخاب می کنیم
۲- نام ، طول و محتوای فایل مورد نظر را به ترتیب برای سرور ارسال می کنیم
۳- در سرور مقادیر را دریافت نموده و فایل را در مسیر جاری برنامه ایجاد می کنیم

برنامه ی کلاینت :



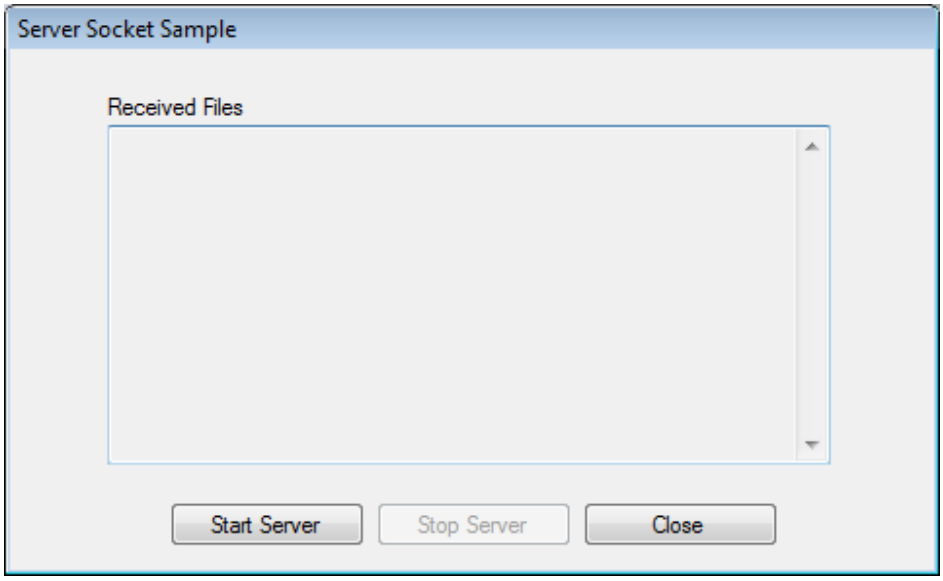
شکل بالا نمایی از کلاینت برنامه است

در برنامه ی کلاینت کد های زیر نوشته می شوند :

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Net;
using System.Net.Sockets;

namespace ClientSocketSample
{
    public partial class Form1 : Form
    {
        private Socket server;
        private string ServerIPAddress = "127.0.0.1";
        private int ServerPortNo = 10000;
        private int MessageLength = 50;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        private void btnClose_Click(object sender, EventArgs e)
        {
            if (server != null)
                server.Close();
            this.Close();
        }
```

```
        private void btnSend_Click(object sender, EventArgs e)
        {
            try
            {
                if (server == null)
                {
                    server = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
                    IPAddress ipAddress = IPAddress.Parse(ServerIPAddress);
                    server.Connect(ipAddress, ServerPortNo);
                }
                // در شرط دستور IF مقدار بازگشتی دکمه های کنترلی شیء
                // openFileDialog را بررسی می نماییم و اگر
                // فایلی انتخاب ، و دکمه ی ok کلیک شده بود
                // جهت ارسال فایل طبق مراحل زیر عمل می کنیم
                if (openFileDialog1.ShowDialog() == DialogResult.OK)
                {
                    string FullFileName = openFileDialog1.FileName;
                    // نام و مسیر فایل را گرفته و در متغیر FullFileName ذخیره نمودیم
                    string FileName = System.IO.Path.GetFileName(FullFileName);
                    // نام فایل را از فایل مربوطه خوانده و در FileName ذخیره می کنیم
                    string FileData = System.IO.File.ReadAllText(FullFileName);
                    // محتوای فایل را از فایل مربوطه خوانده و در FileData ذخیره می کنیم
                    string DataSize = FileData.Length.ToString();
                    // طول فایل را از فایل مربوطه خوانده و در DataSize ذخیره می کنیم
                    txtFileName.Text = FullFileName;
                    byte[] buffer;
                    // در 3 مرحله فایل را ارسال می کنیم
                    buffer = PadMessage(FileName);
                    server.Send(buffer, 0, MessageLength, SocketFlags.None);
                    // نام فایل را برای سرور ارسال کردیم
                    buffer = PadMessage(DataSize);
                    server.Send(buffer, 0, MessageLength, SocketFlags.None);
                    // طول فایل را برای سرور ارسال کردیم
                    buffer = Encoding.ASCII.GetBytes(FileData);
                    server.Send(buffer, 0, buffer.Length, SocketFlags.None);
                    // محتوای فایل را برای سرور ارسال کردیم
                    MessageBox.Show("File sent successfully");
                }
            }
            catch (SocketException)
            {
                server = null;
                MessageBox.Show("Cannot connect to Server");
                return;
            }
        }

        private byte[] PadMessage(string message)
        {
            while (message.Length < MessageLength)
                message += " ";
            byte[] buffer = Encoding.ASCII.GetBytes(message);
            return buffer;
        }
    }
}
```

برنامه ی سرور :



شکل بالا نمایی از سرور برنامه است

در برنامه ی سرور کد های زیر نوشته می شوند :

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

using System.Net;
using System.Net.Sockets;

namespace ServerSocketSample
{
    public partial class Form1 : Form
    {
        private Socket listener;
        private Socket client;
        private string ClientIPAddress = "127.0.0.1";
        private int ClientPortNo = 10000;
        private int MaxConnections = 5;
        private int MessageLength = 50;
        private byte[] ReceivedBuffer;

        public Form1()
        {
            InitializeComponent();
        }

        private void btnClose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
```

```
private void btnStartServer_Click(object sender, EventArgs e)
{
    SetButtonEnabled(btnStartServer, false);
    SetButtonEnabled(btnStopServer, true);
    SetButtonEnabled(btnClose, false);
    listener = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
    EndPoint ipEndPoint = new IPEndPoint(IPAddress.Parse(ClientIPAddress), ClientPortNo);
    listener.Bind(ipEndPoint);
    listener.Listen(MaxConnections);
    AsyncCallback callback = new AsyncCallback(OnAcceprt);
    listener.BeginAccept(callback, null);
}

private void btnStop_Click(object sender, EventArgs e)
{
    if (client != null)
        client.Close();
    listener.Close();
    SetButtonEnabled(btnStartServer, true);
    SetButtonEnabled(btnStopServer, false);
    SetButtonEnabled(btnClose, true);
}

private void OnAcceprt(IAsyncResult ar)
{
    try
    {
        // کانکشن ورودی را که به صورت آسنکرون متصل شده را به سوکت کلاینت نسبت می دهیم
        client = listener.EndAccept(ar);
        // تابع دریافت فایل را فراخوانی می کنیم
        GetFile();
    }
    catch
    {
    }
}
private void GetFile()
{
    while (true)
    {
        try
        {
            ReceivedBuffer = new byte[MessageLength];
            client.Receive(ReceivedBuffer);
            string FileName = Encoding.ASCII.GetString(ReceivedBuffer).Trim();
            // اولین مقدار بافر سوکت را به طول مورد قرارداد دریافت کرده و در FileName قرار دادیم
            ReceivedBuffer = new byte[MessageLength];
            client.Receive(ReceivedBuffer);
            string DataSize = Encoding.ASCII.GetString(ReceivedBuffer).Trim();
            // دومین مقدار بافر سوکت را به طول مورد قرارداد دریافت کرده و در DataSize قرار دادیم
            ReceivedBuffer = new byte[Convert.ToInt32(DataSize)];
            client.Receive(ReceivedBuffer);
            string FileData = Encoding.ASCII.GetString(ReceivedBuffer);
            // سومین مقدار بافر سوکت را به طول فایل ارسالی دریافت کرده و در FileData قرار دادیم
            // حال اگر در مسیر جاری فایل اجرایی سرور فایلی با نام فایل دریافتی وجود
            // داشته باشد را حذف می کنیم
            if (System.IO.File.Exists(FileName))
                System.IO.File.Delete(FileName);
            // فایل دریافتی را در مسیر جاری برنامه ایجاد می کنیم
            System.IO.File.AppendAllText(FileName, FileData);
            // توسط تابع زیر نام فایل دریافتی را در لیست فایلهای دریافتی درج می کنیم
            ShowFileName(FileName);
        }
        catch
        {
            break;
        }
    }
}
```

```
        private void SetButtonEnabled(Button btn, bool enabled)
        {
            System.Threading.ThreadStart todo = delegate
            {
                btn.Enabled = enabled;
            };
            btn.Invoke(todo);
        }

        private void ShowFileName(string FileName)
        {
            System.Threading.ThreadStart todo = delegate
            {
                txtReceivedFiles.Text += FileName + "\r\n";
            };
            txtReceivedFiles.Invoke(todo);
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

    }
}
```