Subject :

Year . Month . Date . ( )

برنامه سازی شبکه

تعریف شبکه های کامپیوتری : یک شبکه کامپیوتری برابر است با اتصال بین یک یا دو یا تعداد بیشتر کامپیوتر یا وسایل جانبی مثل چاپگر، اسکنر.

اهداف شبکه های کامپیوتری : به اشتراک گذاری اطلاعات و فایل ها - به اشتراک گذاری منابع سخت افزاری و نرم افزاری ها

فواید شبکه های کامپیوتری : حذف محدودیت های جغرافیایی و مکانی - افزایش اعتماد - کاهش هزینه - صرفه جویی در وقت.

تقسیم بندی شبکه ها از بعد روش انتقال داده :

۱- **شبکه های نقطه به نقطه :** در این شبکه ها بین هر دو ایستگاه، کانال جداگانه ای وجود دارد بنابراین بین ایستگاه های مختلف مسیرهای متفاوتی وجود خواهد داشت.

۲- **شبکه های پخش همگانی :** یک کانال دارند که بین همه کامپیوترها مشترک است، هرکدام از آنها پیام خود را در بسته هایی بر روی کانال می فرستند و همه ایستگاه ها پیام را دریافت می کنند.
معایب : ظرفیت پایین - مدیریت پیچیده کانال - قابلیت اطمینان پایین کانال.

تقسیم بندی شبکه ها از بعد اندازه :

۱- **LAN :** شبکه های محلی در فواصل جغرافیایی کم (حداکثر چند کیلومتر) مثل اتاق، ساختمان، شرکت و... برقرار می شود.
در این نوع شبکه ها تعداد کامپیوترها کم، هزینه کم و مدیریت شبکه نیز ساده است.
سه پارامتر در این شبکه ها وجود دارد : ۱- اندازه شبکه ۲- فناوری انتقال ۳- توپولوژی
اندازه شبکه ها کم است - فناوری انتقال داده به کابل بستگی دارد، مثلا کابل های UTP در رده های مختلف وجود دارد که هرکدام انتقال های داده ای با نرخ ارسال متفاوت دارند.
توپولوژی یا همبندی چگونگی ارسال اطلاعات توسط کابل های انتقال داده.

۲- **MAN :** این شبکه در مقیاس یک شهر را تحت پوشش قرار می دهد.

Subject:
Year. Month. Date. ( )

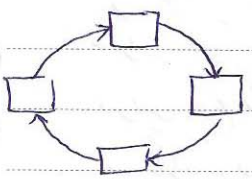
۳_ شبکه WAN : این شبکه شبکه است به ابعاد صدها کیلومتر که محدوده بیشتر در حد یک کشور یا قاره را در بر می گیرد. در این نوع شبکه کامپیوترها به صورت زیر شبکه به هم متصل می شوند و وظیفه زیر شبکه ها ارسال کردن پیام از یک شبکه به شبکه دیگر است.
زیر شبکه ها شامل دو بخش هستند:
۱_ خطوط انتقال
۲_ تجهیزاتی که کار سوئیچ و انتقال داده را انجام می دهند.

۴_ شبکه GAN : شبکه ای است که کامپیوترها و اجزای کامپیوتری سراسر زمین را به یکدیگر متصل می کند. اینترنت نوعی شبکه GAN است و گستردگی آن در حد کره زمین است.

انواع توپولوژی ها:

۱_ BUS : در این توپولوژی تمام ایستگاه ها مستقیما به کانال مشترک متصل اند و تعداد کامپیوترها محدود می باشد. در صورت قطع کانال کل شبکه از کار می افتد.
جذابیت این توپولوژی ارزان است، تعداد کامپیوترها و طول کانال محدود است و خطاهایی در رفع اشکال در این شبکه ها مشکل است.

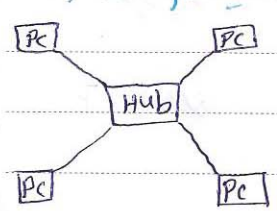
۲_ Ring : در این توپولوژی کامپیوترها به صورت حلقه و نقطه به نقطه به هم متصل اند. انتقال اطلاعات به صورت یکطرفه است. برای ارسال اطلاعات از روش Token Ring (حلقه نشانه) استفاده می شود.



روش Token Ring : در این روش یک کامپیوتر به عنوان مدیر یک Token ایجاد کرده و هر زمان ایستگاهی که Token را در اختیار دارد داده هایش را ارسال می کند.

تعداد کانال ها = تعداد سیستم ها در شبکه

۳_ Star : در این توپولوژی همه کامپیوترها به یک ایستگاه مرکزی به نام مسیریاب یا HUB متصل می شوند و ارتباط آنها از طریق ایستگاه مرکزی برقرار می شود. در صورت خرابی ایستگاه مرکزی کل ارتباط شبکه از کار می افتد.



Subject:
Year. Month. Date. ( )

۴ـ درختی : گسترش یافته توپولوژی ستاره است به طوری که تعدادی HUB به یکدیگر متصل اند و کامپیوترها به HUB متصل اند.

ت ـ mesh : در این توپولوژی هر کامپیوتر مستقیماً از طریق کانال نقطه به نقطه به هر کامپیوتر دیگر درون شبکه اتصال دارد. یک شبکه mesh با n کامپیوتر دارای $\frac{n(n-1)}{2}$ کانال می باشد.

مزایای این توپولوژی عبارتند از:

سرعت انتقال داده ـ قابلیت اطمینان بالا ـ عدم وجود مشکل ترافیک در شبکه

معایب این توپولوژی عبارتند از:

تعداد کامپیوترها زیاد است و هزینه نیز بالاست ـ برپاسازی شبکه مش مشکل و پیچیده است

قابلیت گسترش و اضافه کردن کامپیوترهای جدید به این شبکه مشکل است.

۶ـ Hybrid (ترکیبی):

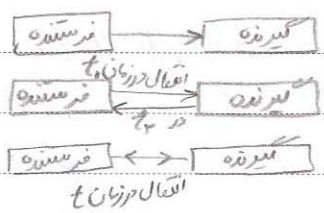
شبکه های بزرگ معمولاً از اتصال چندین توپولوژی مختلف تشکیل شده اند، این توپولوژی بزرگ را توپولوژی ترکیبی می گویند.

انواع تکنولوژی های انتقال داده با توجه به جهت انتقال:

۱ـ کانال یک طرفه simplex مثل تلفن ـ بلوتوث

۲ـ کانال نیمه دوطرفه Half simplex

۳ـ کانال کاملاً دوطرفه Full duplex



مدل مرجع OSI، شرکت Iso:

Iso برای تفکیک وظایف و عملیات لازم تعدادی لایه را معرفی کرده و هر لایه خدماتی را به لایه بالاتر از خود ارائه می دهد.

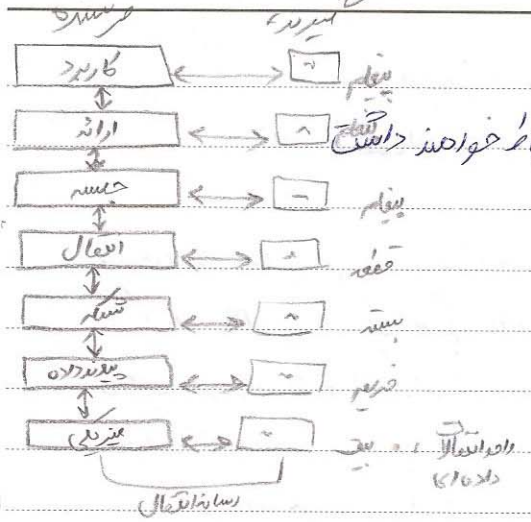
مزایای لایه بندی

۱ـ تغییر دادن هر لایه به دیگر لایه ها تأثیری نگذارد

۲ـ تقسیم شبکه به لایه های کوچکتر حجم آن را ساده تر می کند.

Subject:
Year. Month. Date.

۳- سرعت گسترش و خطا یابی افزایش می یابد.
۴- انواع سخت افزارها و نرم افزارهای مختلف با یکدیگر ارتباط خواهند داشت



وظایف لایه ها:

لایه فیزیکی:

این لایه جهت ارسال بیت ها بر روی کانال ارتباطی است. مشخصات لایه فیزیکی نوع سیگنال های انتقال و نوع رسانه انتقال در این لایه مشخص می شود.

لایه پیوند داده:

این لایه تعیین می کند که توالی ارسال با کدام کامپیوتر است، با کدام سیستم برای کدام سیستم دیگر داده ها را ارسال می کند. مرز ابتدا و انتهای یک فریم را مشخص می کند. کنترل خطا را بر روی اطلاعات انجام می دهد کنترل جریان را بر روی فریم های ارسالی انجام می دهد. مدیریت کانال را بر عهده دارد. آدرس دهی فیزیکی را انجام می دهد. وظیفه تحویل مرتب بسته های داده ای را نیز دارد.

لایه شبکه:

این لایه وظیفه هدایت بسته های اطلاعاتی از مبدا به مقصد را بر عهده دارد.
کنترل ازدحام بر عهده این لایه است
آدرس دهی و مسیر یابی بین فرستنده و گیرنده.

لایه انتقال:

این لایه وظیفه ایجاد، نگهداری و حذف مدار برای انتقال داده را دارد
شکستن و متصل کردن اطلاعات و شمارش گذاری آنها

Subject:
Year. Month. Date. ( )

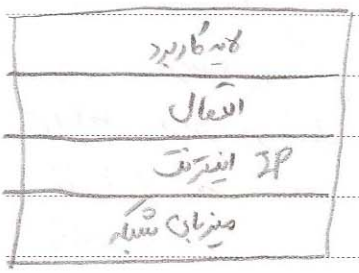
ارائه کیفیت خدمات و کنترل جریان داده : بر عهده این لایه است.

لایه جلسه :
این لایه وظیفه ایجاد، مدیریت و اتمام جلسات بین دو کامپیوتر را دارد
تصدیق هویت فرستنده
اعتبار سنجی پیغام‌ها و همزمان سازی تبادل داده از دیگر وظایف این لایه است.

لایه ارائه : (لایه نمایش) :
وظیفه تبدیل کدهای مختلف داده
رمزگذاری داده در سمت فرستنده و رمزگشایی آن در سمت گیرنده
وظیفه فشرده سازی و از حالت فشرده خارج کردن داده

لایه کاربرد :
این لایه سرویس‌های شبکه‌ای لازم را برای برنامه‌های کاربردی و کاربران فراهم می‌کند.

مدل Tcp/IP
این مدل در شبکه اینترنت استفاده می‌شود ـ ۴ لایه دارد.



مدل TCP/IP و OSI نقاط مشترک زیادی دارند و حدود عملکرد لایه‌هایشان تاحدی شبیه به یکدیگر هستند.
این مدل ۴ لایه داشته و لایه‌های جلسه و نمایش را ندارد.
هر لایه شامل پروتکل‌های مختلف است.

لایه اینترنت :
این لایه شبیه لایه شبکه مدل OSI است و وظیفه بسته‌ها و پروتکل آنها را مشخص می‌کند.
در پروتکل اینترنت شش مرحله ۴ مرحله ارسال و دریافت IP توانایی تکه تکه سازی و پیوند بسته‌های

Subject:
Year. Month. Date. ( )

کوچک شده را دارد و این وظیفه در پروتکل اینترنت شبکه ۶ وجود ندارد.
لایه اینترنت دارای دو عمل اصلی است.
برای بسته های خروجی ایستگاه مقصدی را انتخاب می کند و بسته ها را به آن میزبان منتقل می کند.
برای بسته های ورودی، این لایه بسته های ورودی را دریافت کرده و بسته را به پروتکل مناسب در لایه انتقال منتقل می کند.

لایه انتقال:

در این لایه دو پروتکل TCP و UDP وجود دارد. پروتکل TCP برای انتقالات داده ای اتصال گرا استفاده می شود درحالی که پروتکل UDP پروتکلی بدون اتصال برای انتقالات پیامی است.
پروتکل TCP پروتکل پیچیده ای است زیرا قابلیت اطمینان جریان داده ها را فراهم می کند و نیز TCP کنترل جریان را هم فراهم می کند.
پروتکل UDP امکان انتقال سریع اطلاعات بدون هیچگونه مسئولیتی در رابطه با تضمین صحت اطلاعات را در بر گرفته دارد. به دلیل سریع بودن این پروتکل از این پروتکل برای ارسال تصویر و یا صوت استفاده می شود.

لایه کاربرد:

تمامی برنامه ها و پروتکل های کاربردی در این لایه قرار دارد و با استفاده از این لایه به شبکه دسترسی خواهند داشت. پروتکل های این لایه به منظور فرمت دهی و مبادله اطلاعات کاربران استفاده می شود.
چند نمونه از این پروتکل ها عبارتند از:
HTTP: از این پروتکل برای ارسال فایل ها و صفحات وب استفاده می شود.
FTP: " " ارسال و دریافت فایل استفاده می شود.
DNS: " " " نگاشت اسامی ایستگاه ها به آدرس شبکه آنها استفاده می شود.
تلنت: " " به کاربر اجازه می دهد که ماشین IP مقصدی را راه اندازی کرده و با آن کار کند.

لایه میزبان شبکه:

این لایه مسئول این است که داده ها را بر روی محیط شبکه قرار داده و داده ها را از محیط شبکه دریافت کند.

Subject:
Year. Month. Date. ( )

این لایه شامل دستگاه‌های فیزیکی مثل کابل شبکه است.

تعریف سوکت: مفهومی از تعریف یک ارتباط در سطح برنامه‌نویسی است. برنامه‌نویسی با تعریف سوکت تمایل خود را برای تعریف یک ارتباط به سیستم عامل اعلام کرده و بدون درگیر شدن با جزئیات لایه ۴ام نوع پروتکل مدنظر (TCP / UDP) را مشخص می‌کند. از دید هسته سیستم عامل سوکت یک نقطه انتهایی ارتباط و از دید کاربر ماشه توصیف‌گر فایل می‌باشد که به آن اجازه خواندن و نوشتن به شبکه را می‌دهد. در واقع برنامه‌های سرویس دهنده و سرویس گیرنده می‌توانند از طریق خواندن و نوشتن در سوکت با هم ارتباط داشته باشند.

انواع سوکت:

۱- سوکت استریم (جریان): این نوع سوکت برای ارتباطات اتصال‌گرا استفاده می‌شود و از پروتکل TCP استفاده می‌کند و زمانی این نوع سوکت توصیف می‌شود که صحت داده‌ها اهمیت زیادی داشته باشد.
پروتکل‌های FTP، HTTP از این نوع سوکت استفاده می‌کنند.

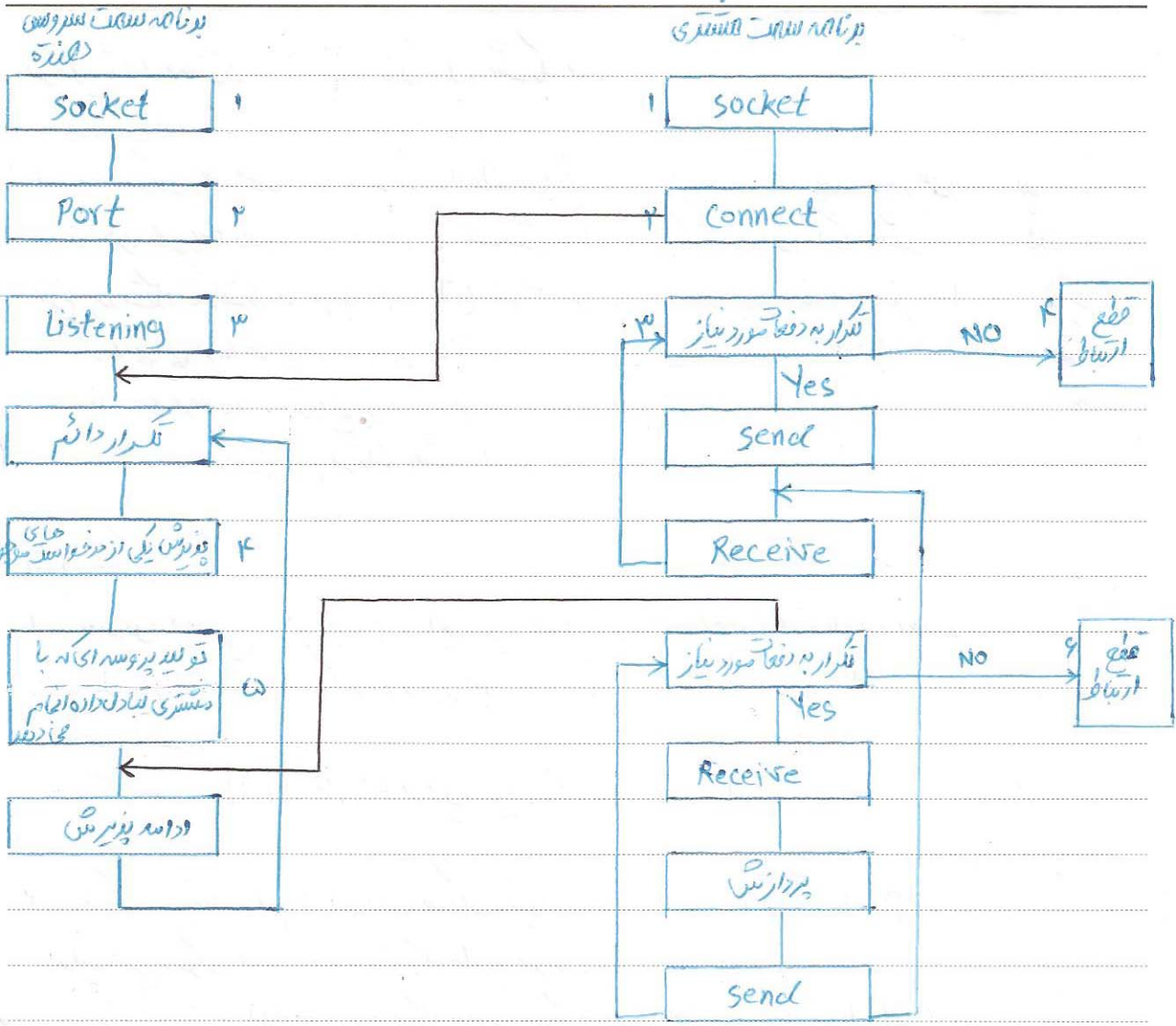
۲- سوکت دیتاگرام: این نوع سوکت غیراتصال‌گرا است و از پروتکل UDP استفاده می‌کند و زمانی از این نوع سوکت استفاده می‌شود که نیاز به سرعت بالا داشته باشد.
پروتکل‌های SNMP (پروتکل مدیریت شبکه) و DNS (پروتکل تبدیل نام) از این نوع سوکت استفاده می‌کنند.

مفهوم کلاینت سرور client/server: سرویس دهنده یا سرور پروسه‌ای است که مقداری اطلاعات در اختیار دارد و با ایجاد یک سوکت نوع دیتاگرام یا استریم قصد دارد این اطلاعات را به اشتراک بگذارد.

مشتری یا client هم پروسه‌ای است که نیازمند مقداری اطلاعات است و با ایجاد یک سوکت نوع استریم یا دیتاگرام قصد دارد اطلاعات به اشتراک گذاشته شده توسط سرویس دهنده را به دست آورد.
طرفین ارتباط حتماً باید سوکت تعریف کنند و نوع سوکت تعریف شده توسط طرفین ارتباط باید یکسان باشد.

Subject:

Year. Month. Date. ( )



برنامه سمت سرویس دهنده :

۱) socket : سرویس دهنده یک سوکت ایجاد کرده و به این ترتیب به سیستم عامل اعلام می کند که قصد تعریف یک ارتباط را دارد در همین مرحله نوع پروتکل لایه انتقال TCP یا UDP را مشخص می کند.

۲) port : به سوکتی که باز کرده یک شماره پورت اختصاص می دهد با این کار اعلام می کند که پذیرنده تعدادی بسته است.

۳) listening : به سیستم عامل اعلام می کند که عملیات دریافت بسته های مربوط به یک پروسه را آغاز کند. در این مرحله مشخص می کند که حداکثر چقدر ارتباط را پشتیبانی می کند.

۴) از سیستم عامل تقاضا می کند که یکی از ارتباط های موجود را به آن تحویل دهد.

Subject:
Year. Month. Date. ( )

د) با استفاده از توابعی مانند Send و Receive اقدام به تبادل داده می‌کند.
ه) به صورت هماهنگ با پروسه مشتری یا به صورت یک طرف ارتباط برقرار شده را قطع می‌کند.

برنامه سمت مشتری:
۱) socket: یک سوکت را که مشخص کننده یک ارتباط است را ایجاد می‌کند و به سیستم عامل اعلام می‌کند که قصد تعریف یک ارتباط را دارد.
۲) connect: برخلاف سرویس دهنده که شماره پورت ثابت را به خود اختصاص می‌دهد، در این مرحله با استفاده از یک شماره پورت تصادفی به مقصد متصل می‌شود.
۳) شروع به تبادل داده می‌کند (با استفاده از دستورات Send و Receive).
۴) ارتباط برقرار شده را قطع می‌کند و منابع را آزاد می‌کند.

پروتکل تلنت Telnet: یک پروتکل ارتباطی است و استفاده از آن مربوط به زمانی است که کامپیوتر شخصی وجود نداشته باشد.
برنامه Telnet یک ترمینال مجازی و سازگار و با ساخت افزار موجود برای ارتباط با سرویس دهنده Telnet شبیه سازی می‌کند. این پروتکل یک پروتکل متنی است و با ارتباط با سرویس هایی که خدمات فنی ارائه می‌دهند قابل استفاده است، مانند HTTP و SMTP.

نشست Telnet: منظور از نشست Telnet برقراری موفقیت آمیز یک اتصال با پورت 23 سرویس دهنده است که برای این اتصال از پروتکل Tcp استفاده می‌شود.
نشست Telnet زمانی آغاز می‌شود که مشتری هویت خود را به سرور اثبات کند، پس از برقراری نشست Telnet مشتری دستور خود را از طریق صفحه کلید وارد کرده، برنامه Telnet کلیدهای فشرده شده را در قالب یک بسته Tcp به سمت پورت 23 سرور ارسال می‌کند. سرویس دهنده Telnet در صورت معتبر بودن دستور دریافت شده محاسبات را انجام داده و پاسخ را در قالب یک بسته Tcp برای مشتری برمی‌گرداند که در صفحه نمایش مشتری نشان داده می‌شود.

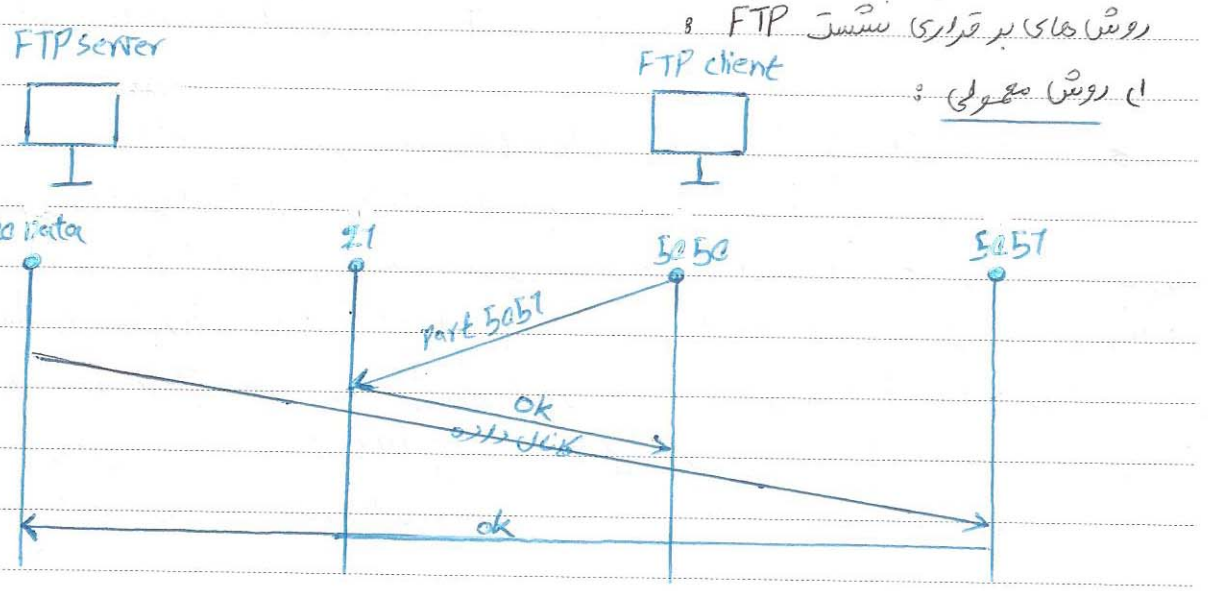
Subject:
Year. Month. Date. ( )

پروتکل FTP : این پروتکل یک پروتکل قدرتمند برای انتقال فایل در طول شبکه است که در لایه انتقال از پروتکل TCP استفاده می کند. بنابراین از سوکت نوع استریم استفاده می کند.
با استفاده از پروتکل FTP می توان عملیات زیر را انجام داد:
۱- تهیه فهرستی از فایل های موجود در یک پوشه بر روی سرور
۲- حذف و تغییر نام و جابه جا کردن فایل ها و پوشه های روی سرور
۳- ایجاد یا حذف پوشه بر روی سرور از راه دور
۴- بارگذاری فایل بر روی سرور FTP از راه دور
۵- بارگیری فایل از روی سرور FTP از راه دور

* برای ایجاد یک نشست FTP بین مشتری و سرویس دهنده باید دو اتصال TCP همزمان بین آن ها برقرار شود:
۱) کانال داده = یک اتصال TCP بین پروسه مشتری و پورت شماره ۲۰ سرویس دهنده.
۲) کانال فرمان = یک اتصال TCP ~ ~ ~ ~ ~ ~ ۲۱ ~ .

* پروتکل FTP مشتری و سرویس دهنده هرکدام دو سوکت از نوع استریم ایجاد می کنند

روش های برقراری نشست FTP :
۱) روش معمولی :



مرحله اول : پروسه مشتری دو سوکت نوع استریم ایجاد کرده و شماره پورت های تصادفی و بالای ۱۰۲۴ به آنها نسبت می دهند.

مرحله دوم : مشتری با استفاده از یکی از سوکت ها درخواست اتصال را به سمت پورت ۲۱ سرویس دهنده FTP ارسال می کند.

مرحله سوم : مشتری با استفاده از یکی از این ارتباطات شماره پورت خود را به اطلاع سرویس دهنده می رساند.

مرحله چهارم : پروسه سرویس دهنده با سوکتی که پورت ۲۰ را به آن اتصال داده به پورتی از مشتری وصل می شود که در مرحله قبل به او گفته شده.

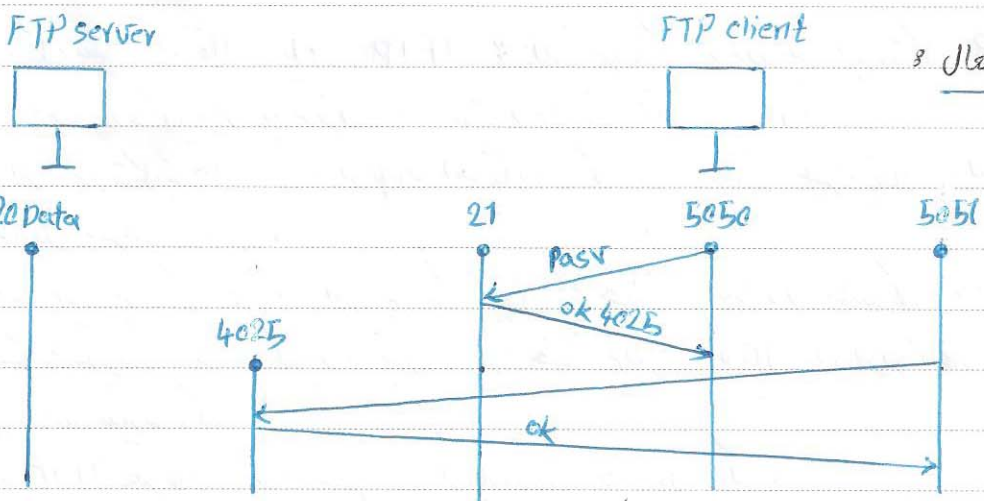
مرحله پنجم : مشتری تقاضای اتصال سرویس دهنده را پذیرفته و نشست FTP آغاز می شود.

دلایل استفاده از دو سوکت به طور همزمان :

۱- برای اینکه تبادل داده ها بدون قطع جریان صورت بگیرد از پورت متمایز استفاده می شود.

۲- به منظور عدم استفاده از کدگذاری های متفاوت برای فرمان و داده و ارسال آنها از طریق یک کانال از دو پورت متمایز به طور همزمان استفاده می شود.

۲) روش غیر فعال :



مرحله اول : مشتری دو سوکت نوع استریم باز کرده و دو شماره پورت تصادفی بالای ۱۰۲۴ به آنها نسبت می دهد.

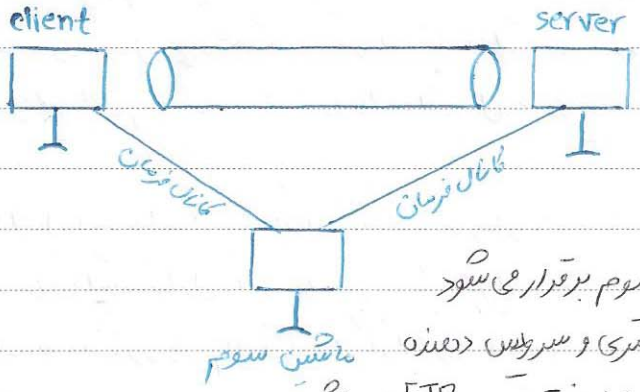
مرحله دوم : با استفاده از یکی از سوکت های باز شده به پورت ۲۱ سرور FTP متصل می شود

Subject:
Year. Month. Date. ( )

مرحله سوم: مشتری با ارسال فرمان PASV به سمت سرور اعلام می‌کند که قصد برقراری یک نشست PASV (غیرفعال) را دارد.

مرحله چهارم: پروسه سرویس دهنده یک سوکت جدید با شماره پورت تصادفی بزرگتر از ۱۰۲۴ ایجاد کرده و این شماره پورت را به مشتری اعلام می‌دهد.

مرحله پنجم: مشتری با استفاده از سوکت خود به پورتی از سرویس دهنده متصل می‌شود که مرحلهٔ قبل به او اعلام شده است.



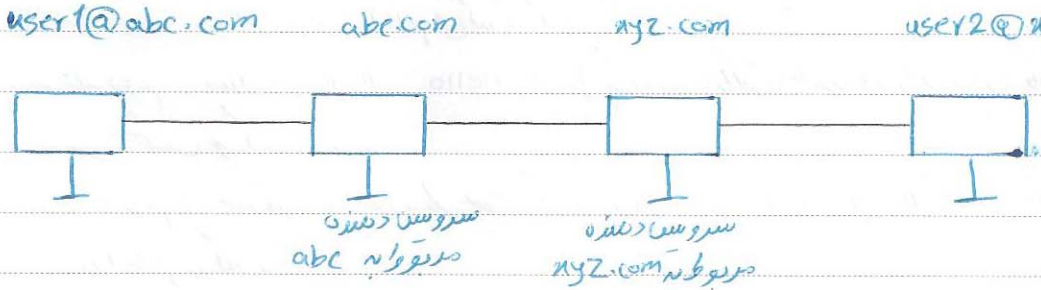
انتقال با واسطه در پروتکل FTP: در این پروتکل این امکان وجود دارد که نظارت بر انتقال داده‌ها توسط یک ماشین سوم انجام گیرد.
در این حالت کانال فرمان از طریق یک ماشین سوم برقرار می‌شود اما کانال داده مستقیماً بین پروسه های مشتری و سرویس دهنده برقرار می‌شود. هدف از انجام این کار بالابردن امنیت سرور FTP می‌باشد.

پروتکل ساده انتقال فایل TFTP: این پروتکل در دو زمینه با پروتکل FTP تفاوت دارد:
۱- در این پروتکل نیازی به برقراری نشست و عملیات لاگین وجود ندارد.
۲- این پروتکل به جای TCP از UDP استفاده می‌کند که یک پروتکل غیرانتقال‌گرا است و موجب بالابردن سرعت می‌شود.

معایب این پروتکل: ۱- این پروتکل نمی‌تواند خطاهای ساده‌ای مانند کمبود حافظه یا فضای دیسک را مدیریت و برطرف کند و با بروز هرگونه خطا عملیات انتقال باید از نو آغاز شود و مثل FTP نمی‌تواند مدیریت کند.
۲- TFTP هر فایلی را که سیستم عامل اجازه بدهد را منتقل می‌کند.

Subject:
Year. Month. Date. ( )

پروتکل SMTP برای انتقال نامه‌های الکترونیکی:



فرض کنید کاربر user1@abc.com قصد ارسال نامه به کاربر user2@xyz.com را داشته باشد. کاربر user1 ابتدا نامه را به سمت سرویس دهنده SMTP مربوط به حوزه abc.com ارسال می‌کند سپس سرور مربوط به حوزه abc.com پس از شخص شدن قصد نامه آن را به سمت سرور SMTP مربوط به حوزه xyz.com ارسال می‌کند. این سرور نامه را در فضایی که به کاربر user2 تخصیص داده شده ذخیره می‌کند.

تبادل نامه بین دو کاربر: کاربر ابتدا یک اتصال TCP با پورت ۲۵ سرویس دهنده مربوط به حوزه خود برقرار می‌کند بعد مراحل تبادل نامه به صورت زیر است:

1) 250 xyz.com SMTP service Ready
2) Hello abc.com
3) 250 xyz.com says hello to abc.com
4) MAIL from: <user1@abc.com>
5) 250 sender ok
6) To: <user2@xyz.com>
7) 250 Receipt ok
8) DATA
9) send mail; end with "." on a line by itself.
10) ارسال نامه
11) 250 message accepted
12) Quit
13) xyz.com closing connection

Subject:
Year. Month. Date. ( )

توضیح مرحله اول: سرویس دهنده به محض ارتباط TCP یک کد به همراه یک رشته کاراکتری برای مبدأ نامه ارسال می‌کند و به این ترتیب اعلام می‌کند که آمادگی دارد.

مرحله دوم: مبدا با ارسال کلمه Hello و ارسال یک رشته کاراکتری حوزه‌ای را که به آن تعلق دارد را برای سرور مشخص می‌کند.

مرحله سوم: سرور نامه حوزه را بررسی کرده و در صورت قابل ارسال یک کد مشخص آمادگی خود را اعلام می‌کند.

مرحله چهارم: مبدا با ارسال فرمان MAIL FORM فرستنده نامه را مشخص می‌کند.

مرحله پنجم: سرور صاحب نامه را بررسی کرده و در صورتی که منعی برای دریافت نامه از این کاربر وجود نداشته باشد به کاربر اعلام آمادگی می‌کند.

مرحله ۶: مبدا گیرنده نهایی نامه را به سرور اعلام می‌کند.

مرحله ۷: سرور گیرنده نامه را بررسی کرده و در صورتی که منعی برای این کار وجود نداشته باشد آمادگی خود را برای دریافت نامه اعلام می‌کند.

مرحله ۸: مبدا با ارسال کلمه DATA آمادگی خود را برای ارسال اعلام می‌کند.

مرحله ۹: سرور ضمن اعلام آمادگی از کاربر درخواست می‌کند که در آخرین خط نامهٔ خود فقط یک نقطه قرار دهد تا پایان نامه را متوجه شود.

مرحله ۱۰: مبدأ نامهٔ خود را ارسال می‌کند.

مرحله ۱۱: سرور دریافت موفقیت آمیز نامه را به اطلاع مبدأ می‌رساند.

مرحله ۱۲: مبدأ با ارسال کلمه QUIT اعلام خروج می‌کند.

مرحله ۱۳: ارتباط TCP بسته می‌شود.

برای برنامه نویسی شبکه به زبان C به موارد زیر نیاز داریم:

۱- یک کامپایلر مناسب برای زبان مربوطه
۲- فایل های سرآیند مربوط به برنامه نویسی شبکه
۳- سیستم عاملی که از سوکت های شبکه پشتیبانی می‌کند.

برای برنامه نویسی شبکه در ویندوز به فایل سرآیند <winsock.h> نیاز مندیم.

Subject:

Year. Month. Date. ( )

توابع مهم موجود در فایل سرآیند winsock.h برای برنامه نویسی تحت شبکه :

۱- تابع wsatartup : این تابع برای آماده سازی و بارگزاری اولیه سیستم عامل برای اجرای برنامه تحت شبکه مورد استفاده قرار می گیرد. نگارش این تابع به صورت زیر است :

int wstartup (word wversion Requested , WSADATA wsa data);

نوع — پارامتر اول — نوع ساختمان — پارامتر دوم — نام ساختمان

پارامتر اول دارای نوع شماره نسخه فایل سرآیند winsock.h را مشخص می کند که برای تبدیل این شماره به نوع word از تابع wakeword استفاده می کند.

version 1,۲ : makeword(1,۲)

پارامتر دوم تابع wstartup یک متغیر از نوع ساختمان WSADATA است که این نوع ساختمان در فایل سرآیند winsock.h تعریف شده است.

تعریف ساختمان در زبان C :

Tpedef struct struct_name {

نوع ساختمان — ساختمان — نام ساختمان

انواع مختلف داده (int , double , char ...)

```
int a;
char name [15];
char family [25];
double ave;
⋮
};
```

با توجه به تعریف ساختمان می خواهیم ساختمان WSADATA را بنویسیم :

```
Typedef struct WSADATA{word wversion;
word wHeighversion;
char system status [اندازه];
char Description [اندازه];
int imax sockets;
int imax udpDg;
⋮
};
```

Subject:
Year. Month. Date. ( )

WordVersion : در این متغیر شماره نسخه سوکت قرار می‌گیرد.

wordWHeighVersion : بالاترین نسخه از ویندوز سوکت را مشخص می‌کند.

char systemStatus : یک رشته است که اطلاعات مربوط به پیکربندی سیستم در آن قرار می‌گیرد

char Description : از نوع کاراکتری است و توصیفی از پیاده‌سازی سوکت ویندوز را در آن قرار می‌دهند.

int imaxSockets : بیشترین تعداد سوکت را مشخص می‌کند که می‌تواند ایجاد و باز شود.

int imaxUdpDg : بیشترین اندازه یک پیغام دیتاگرام را مشخص می‌کند.

دیتاگرام

نحوه استفاده از تابع WSAStartup :

```
WSADATA wsa;
WSAStartup(makeword(2,2),&wsa);
```

WSAStartup(wVersionRequested, &wsa)

این تابع در صورت موفقیت مقدار 0 و در صورت شکست مقدار یک را برمی‌گرداند.

تابع WSACleanup :

```
int WSACleanup(void);
```

این تابع برای اتمام بارگزاری مربوط به شبکه در سیستم عامل است که با اجرای آن منابع اختصاص داده شده آزاد می‌شود. این تابع در صورت موفقیت مقدار 0 و در صورت بروز خطا مقدار 1 را برمی‌گرداند

تابع socket :

نوع تابع — اسم تابع

```
SOCKET socket(int af, string type, Protocol);
```

این تابع برای تعریف و ایجاد یک سوکت به کار می‌رود و مقدار بازگشتی آن توصیف کننده سوکت مورد نظر است.

af : این پارامتر مشخص کننده نوع آدرس است

type : این پارامتر نوع ارتباط را تعیین می‌کند که برای ارتباط Tcp از ثابت sockt_stream و برای ارتباط UDP از ثابت sock_DGRAM استفاده می‌کند.

Subject:
Year. Month. Date. ( )

protocol: این پارامتر نشان دهنده پروتکل انتخابی در لایه انتقال است که برای ارتباط Tcp از ثابت IPPRoto_Tcp و برای ارتباط UDP از ثابت IPProto_UDP استفاده می کند.

تابع bind (انتساب دادن): *name

int bind (socket s, struct socketaddr *name, int name_len)

این تابع وظیفه پیوند دادن اطلاعات ارتباطی مانند آدرس و پورت به سوکت تعریف شده را بر عهده دارد.
این تابع فقط در سمت سرور مورد استفاده قرار می گیرد.

پارامترهای تابع:

1. socket s — پارامتر اول این تابع متغیر نوع سوکت تعریف شده در برنامه را مشخص می کند.
2. struct socketaddr *name — پارامتر دوم آدرس محلی از حافظه است که متغیر ساختمان در آنجا تعریف شده است

** ساختمان برای نگهداری Data های مختلف استفاده می شود **

متغیر ساختمان sock_addr:

```
struct sock_addr {
int sin_port
int sin_addr
}
```

sin_port: مشخص کننده شماره پورت ارتباطی است.

sin_addr: برای نگهداری آدرس IP است.

3. int name_len — پارامتر سوم اندازه ساختمان واقع در پارامتر دوم را مشخص می کند.

برای تعیین اندازه ساختمان از تابع sizeof() استفاده می کند.

```
bind (s, sockaddr &recIP, sizeof (recIP));
```

s: اسم سوکت
&recIP: آدرس ساختمانی که حاوی آدرس و شماره پورت است.
sizeof(recIP): اندازه ساختمان

این تابع محافظ اطلاعات درون recIP (آدرس و شماره پورت) را به سوکت s اختصاص می دهد.

Subject:
Year. Month. Date. ( )

تابع Listen :

```
int listen (socket s , int backlog );
```

تابع Listen وظیفه گوش دادن به خط را در برنامه server بر عهده دارد و از سیستم عامل می‌خواهد که دریافت بسته‌ها و اتصالات مربوط به آن را آغاز کند.

پارامترهای تابع :

1. s : socket و متغیر نوع سوکت تعریف شده در برنامه است.
2. backlog : تعداد کانکشن‌های درخواستی است که در یک لحظه به سیستم سرویس دهنده درخواست داده‌اند و سیستم می‌تواند آن‌ها را در حالت معلق نگه دارد.

تابع accept :

```
socket accept (socket s , struct sockaddr * address , int address_len)
```

این تابع مانند تابع bind است با این تفاوت که در پارامتر دوم مشخصات سیستم متصل شده به سرور قرار می‌گیرد. اگر سرور نیازی به اطلاعات ماشین متصل شده به آن نداشته باشد می‌توان به جای پارامتر دوم و سوم از مقدار null استفاده کرد.

تابع connect :

```
int connect (socket s , struct sockaddr *name , int name_len)
```

این تابع وظیفه برقراری ارتباط با پروسه server را بر عهده دارد.

1. s : socket : سوکت تعریف شده در برنامه را مشخص می‌کند
2. struct sockaddr : یک ساختمان است که در بر دارنده اطلاعات ارتباطی مربوط به سرور است
3. name_len : طول ساختمان مشخص شده در پارامتر دوم را مشخص می‌کند.

تابع send :

```
int send (socket s , char * buffer , int len , int flags )
```

این تابع وظیفه ارسال بسته را انجام می‌دهد.

Subject:
Year. Month. Date. ( )

پارامترهای تابع :

1. Socket s : سوکت تعریف شده در تابع را مشخص می‌کند.
2. char* buffer : آدرس محلی از حافظه است که داده‌های ارسال شونده در آنجا قرار گرفته.
3. Len : طول اطلاعات ارسال شونده برحسب بایت است.
4. Flags : شامل شوایتی برای تنظیمات ارسال است و در حالت معمول برابر 0 است

تابع receive :

```
int receive (socket s , char* buffer , int len , int flags)
```

این تابع وظیفه دریافت بسته را به عهده دارد.
پارامترهای این تابع مشابه تابع send است با این تفاوت که در پارامتر دوم آدرس محلی از حافظه را مشخص می‌کنیم که داده‌های دریافتی در آنجا ذخیره خواهند شد.
این دو تابع (یعنی تابع send و receive) برای ارتباطاتی مورد استفاده قرار می‌گیرند که بر مبنای پروتکل TCP است.

تابع sendto :

```
int sendto(SOCKET s,char* buffer,int len,int flags,struct sockaddr
,int tolen);
```

این تابع مشابه تابع send است با این تفاوت که اطلاعات ارتباطی بروسه معتبر نیز برای آن مشخص می‌شود چون در ارتباطات UDP از این تابع استفاده شده و UDP یک پروتکل غیر اتصال گرا است مقدار بازگشتی توابع send و receive تعداد بایت‌های ارسالی یا دریافتی است.

تابع recvFrom :

```
int recvFrom (SOCKET s, char* buffer, int len , int flags, struct sockaddr
, int fromlen);
```

این تابع مشابه تابع receive می‌باشد. در ارتباطات UDP از این تابع استفاده می‌شود. پارامترهای این تابع مشابه sendto است.

Subject:
Year. Month. Date. ( )

تابع shutdown:

```
int shutdown (socket s, int how);
```

پارامتر how برای مشخص کردن نوع رفتاری است که این تابع از خود نشان می دهد. یعنی نحوه shutdown شدن سوکت مورد نظر توسط این پارامتر تعیین می شود.

مقادیر مختلف پارامتر how:

۱) SD_Receive یا عدد صفر: عملیات ارسالی نخواهیم داشت اما عملیات دریافت همچنان ادامه می یابد.

۲) SD_send یا عدد یک: عملیات دریافت خاتمه یافته اما عملیات ارسال می تواند ادامه داشته باشد.

۳) SD_Both یا عدد دو: هردو عملیات ارسال و دریافت خاتمه می یابد.

تابع close socket:

```
int closesocket(SOCKET s);
```

این تابع سوکت s را آزاد می کند.

لایه شبکه

اتصال چندین شبکه به یکدیگر و ایجاد یک شبکه بزرگتر را شبکه بندی گویند.

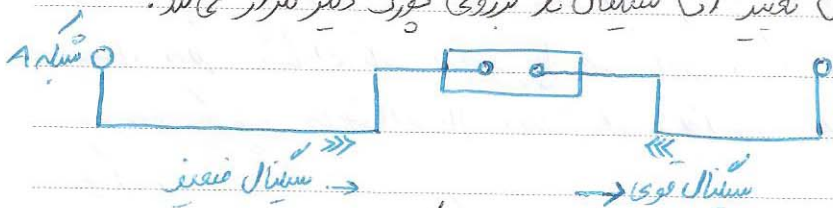
تجهیزات سخت افزاری اتصال دهندهٔ شبکه ها به یکدیگر

۱. تکرارگر (Repeater):

برای اتصال دو شبکه با ۷ لایه یکسان OSI از تکرارگر استفاده می شود. تضعیف یکی از مشکلات مشترک رسانه های انتقال است. افت این میزان حداکثر فاصله برای انتقال داده توسط رسانه انتقال

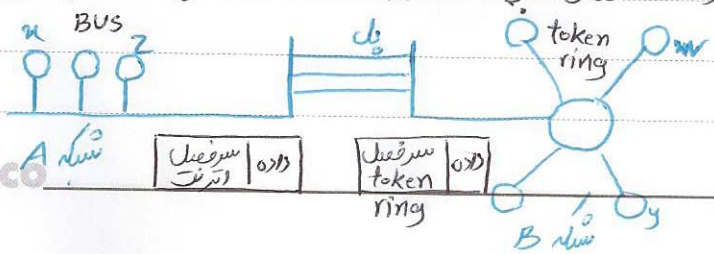
Subject:
Year. Month. Date. ( )

دیل استفاده از تکرارگر است. تکرارگر دستگاهی است که سیگنال دریافتی از یک پورت را قبل از اینکه تضعیف شود و بدون تغییر آن سیگنال را بروی پورت دیگر تکرار می‌کند.



تکرارگر هیچ فیلتری بروی داده انجام نمی‌دهد و تمام ترافیک را منتقل می‌کند.
یک تکرارگر در لایه‌ی اول مدل OSI یعنی لایه فیزیکی عمل می‌کند.
تکرارگر نمی‌تواند ۲ شبکه با پروتکل‌های مختلف را به یکدیگر متصل کند اما بعضی از تکرارگرها قابلیت اتصال دو شبکه با پروتکل یکسان ولی با رسانه‌ی انتقال متفاوت را دارند.
تجهیزات HUB و MAU نوعی تکرارگر هستند. از HUB در اترنت و از MAU در token ring استفاده می‌شود. از طریق چندین HUB و MAU می‌توان چندین شبکه با پروتکل یکسان را به یکدیگر متصل نمود.
۳ نوع HUB به صورت زیر وجود دارد:
۱_ HUB غیرفعال: این هاب فقط ارتباط بین کامپیوترها را برقرار می‌کند.
۲_ HUB فعال: این هاب سیگنال دریافتی را دوباره تولید و ارسال می‌کند بنابراین فاصله‌ی بین کامپیوترها می‌تواند افزایش یابد. این دو نوع هاب (فعال و غیرفعال) در لایه فیزیکی مدل OSI عمل می‌کنند.
۳_ switch HUB: هاب‌های فعال و غیرفعال با دریافت یک فریم آن را بروی تمام پورت‌های دیگر تکرار می‌کند اما سوئیچ‌ها با دریافت یک فریم و بررسی آدرس مقصد آن فریم را فقط بروی پورت متصل به مقصد تکرار می‌کند. این نوع هاب در لایه دوم مدل OSI یعنی لایه پیوند داده عمل می‌کند.

2. پل (Brielge): پل در لایه دوم مدل OSI عمل می‌کند. دو شبکه حتی اگر لایه فیزیکی و پیوند داده متفاوت داشته باشند اما در دیگر لایه‌ها یکسان باشند را می‌توان توسط پل به یکدیگر متصل کنیم. مثلاً دو شبکه که یکی از پروتکل token ring با رسانه‌ی انتقال UTP استفاده می‌کند را می‌توان به شبکه دیگری که از پروتکل اترنت و رسانه انتقال کابل coaxial استفاده می‌کند متصل نمود.



Subject:
Year. Month. Date. ( )

مراحل کار پل به صورت زیر است :
وقتی یک پورت پل فریمی را دریافت میکند تصمیم می گیرد که فریم را حذف کند (اگر آدرس مقصد آن متعلق به همان شبکه ای باشد که پورت پل فریم را دریافت کرده) و یا آن را به سمت شبکه دیگری و بر روی دیگر پورت ها شبیه ارسال دهد (با توجه به آدرس مقصد برای پورت مناسب ارسال می کند.

دلایل استفاده از پل ها در شبکه محلی :
۱- گسترش جغرافیایی
۲- ایجاد شبکه های مختلف
۳- عدم انتقال ترافیک محلی
مزیت اصلی پل نسبت به تکرار گر عدم انتقال ترافیک محلی است .
تکرار گر بدون اطلاع از وضعیت فریم نوع ترافیک عبوری آن را منتقل می کند . اما پل به علت عدم انتقال ترافیک محلی مزایای زیر را فراهم می کند :
۱- افزایش سرعت
۲- افزایش فاصله جغرافیایی
۳- افزایش امنیت

سوئیچ . مسیریاب ها (روتر) :
برای انتقال شبکه های محلی به اینترنت از مسیریاب استفاده می شود . انتقال بسته های اطلاعاتی بین فرستنده و گیرنده با توجه به آدرس های منطقی را مسیریابی می نامند که این وظیفه بر عهده مسیریاب است .
از مسیریاب می توان جای پل استفاده کرد ولی عکس آن امکان پذیر نیست .
مسیریاب می تواند بسته ها را برای شبکه های مختلف ترجمه و تبدیل کند اما پل در لایه پیوند داده عمل می کند و به پیوند داده عبوری به لایه فیزیکی وابسته است . مسیریابها اجازه ارسال داده به صورت بخش همگانی را فراهم نمی کنند و باید حتما آدرس مقصد مشخص و یکتا باشد .
در شبکه ها از دو نوع مسیریاب استفاده می شود که عبارتند از :
۱- مسیریابهای ایستا :

Subject:
Year. Month. Date. ( )

این مسیریابها نمی‌توانند مسیرهای مشخص کنند و باید جدول مسیریابی آنها به صورت دستی پیکربندی شود.

۲- مسیریاب پویا:

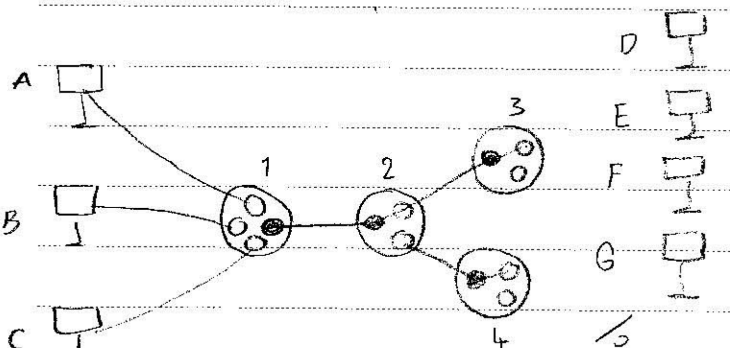
این مسیریابها با توجه به اطلاعات بسته‌ها (آدرس مبدا و مقصد) و همچنین اطلاعات دریافت شده از دیگر مسیریابها می‌توانند شبکه‌ها را بشناسند و براساس آنها می‌توانند مسیر بهینه‌ای را تعیین کنند.

4. دروازه‌ها : Gateway

دروازه به معنی عملکرد یک سیستم در لایه‌های بالای مدل OSI است و برای ارتباط بین سیستم‌های با معماری و پروتکل‌های متفاوت است. مثلا تبدیل معماری ۷ لایه‌ای به معماری ۴ لایه‌ای TCP بر عهده دروازه است.

دروازه دو محیط مختلف را به یکدیگر متصل کرده به طوری که سرفصل یک بسته دریافت شده براساس یک پروتکل خاص را حذف نموده و با سرفصل پروتکل دیگر جایگزین می‌کند. دروازه در لایه ۷ام OSI (لایه کاربردی) عمل می‌کند. اگرچه در تمام لایه‌های مدل OSI نیز می‌تواند عمل کند.

سوئیچینگ : switching



زمانیکه چندین کامپیوتر از طریق کانال به یکدیگر متصل شوند دو روش استفاده از کانال پخش همگانی یا کانال نقطه به نقطه وجود دارد. زمانی که تعداد کامپیوترها بسیار زیاد و یا شبکه بسیار بزرگ باشد استفاده از کانال پخش همگانی و توپولوژی نقطه به نقطه امکان پذیر نیست. راه بهتر استفاده از روش سوئیچینگ است. یک شبکه سوئیچ از تعدادی گره متصل به هم به نام سوئیچ تشکیل شده. سوئیچ‌ها دستگاه‌های سخت افزاری هستند که ارتباط موقت میان دو یا چند کامپیوتر متصل به سوئیچ را فراهم می‌کنند. در شبکه سوئیچ مسیریابها به عنوان سوئیچ و برای مسیریابی به کار می‌روند.

Subject:
Year. Month. Date. ( )

سه روش مختلف سوئیچینگ وجود دارد:

۱- سوئیچینگ مداری

۲- " پیغامی

۳- " بسته ای

سوئیچینگ مداری: برای ایجاد یک مدار اختصاصی و مسیر فیزیکی واقعی بین فرستنده و گیرنده از روش سوئیچینگ مداری در لایه فیزیکی استفاده می شود.

این مدار فقط مختص فرستنده و گیرنده است. در این روش داده به صورت جریانی از بیت ها و بدون نیاز به بسته بندی و قرار دادن آدرس مبدا و مقصد در مدار فیزیکی اختصاصی بین دو کامپیوتر منتقل می شود.

این روش دارای ۳ مرحله برقراری ارتباط بین فرستنده و گیرنده - مرحله انتقال داده - مرحله قطع ارتباط است.

سوئیچینگ پیغام: در این روش کانالی بین حد کامپیوتر و مرکز سوئیچ موجود است و حد کامپیوتر هر لحظه که نیاز به ارسال داشت اطلاعات را در غالب یک پیغام شامل سرفصل و رشته داده ارسال می کند و مرکز سوئیچ ابتدا کل پیغام را دریافت نموده و در حافظه خود ذخیره نموده و با توجه به آدرس مقصد پیغام را به سمت خروجی مناسب و مرکز سوئیچ بعدی ارسال می کند تا در نهایت مقصد پیغام را دریافت کند.

در این روش مانند روش سوئیچینگ مداری نیازی به برقراری ارتباط مداری بین فرستنده و گیرنده نیست بنابراین زمان برقراری ارتباط حذف می شود اما با روش ذخیره و هدایت در سوئیچینگ پیغام سرعت در دریافت بسیار پایین تر از روش سوئیچینگ مداری است.

ترتیب دریافت بسته ها در سوئیچینگ پیغام حفظ نمی شود اما سوئیچینگ مداری ترتیب را حفظ می کند.

سوئیچینگ بسته:

در سوئیچینگ بسته ای داده ها توسط بلوک های با طول متفاوت به نام بسته به صورت گسسته در کانال انتقال می یابد. حداکثر طول بسته ها توسط شبکه معین می شود. در سوئیچ با دریافت

Subject:
Year. Month. Date. ( )

کامل بسته می تواند آن را هدایت کند. درحالی که به طور همزمان می تواند بسته دیگری را دریافت کند. زمان ارسال و دریافت بسته ها روی هم افتاده و تاخیر نسبت به سوئیچینگ پیغامی کاهش می یابد. بسته ها در طول شبکه از یک سوئیچ به سوئیچ دیگر منتقل می شوند و در هر سوئیچ ابتدا ذخیره می شوند سپس بررسی شده تحلیل آن و جدول مسیریابی به سمت مناسب هدایت می شوند. دو روش سوئیچینگ بسته عبارتند از 1 مدار مجازی 2 داده گرام.

سوئیچینگ بسته ای مدار مجازی:

این روش دارای سه مرحله است:

۱. در ابتدا و در مرحله برقراری ارتباط یک مسیر مجازی از طریق سوئیچ ها بین فرستنده و گیرنده در لایه نیز یکی برقرار می شود و با توجه به آدرس مبدا و مقصد یک شناسه مجزا به فرد برای هدایت بسته هایی با همان شناسه در جداول سوئیچ های واقع در مسیر مجازی ایجاد می شود.

۲. در مرحله انتقال داده فرستنده حتی گام ارسال اطلاعات به هر بسته همان شناسه مرحله برقراری ارتباط را نسبت می دهد و سپس ارسال می کند. بسته ها با توجه به شناسه از سوئیچ ها می گذرند و به سمت مقصد هدایت می شوند.

۳. در نهایت پس از اتمام عمل انتقال داده در مرحله قطع ارتباط مسیر مجازی بین فرستنده و گیرنده از بین می رود. قسمتی از این مدار ممکن است برای دیگر فرستنده ها و گیرنده ها مورد استفاده قرار گیرد. در این روش ترتیب دریافت بسته ها توسط گیرنده حفظ می شود.

سوئیچینگ بسته ای داده گرام:

در این روش اطلاعات فرستنده به چندین بسته تقسیم می شود و هر بسته علاوه بر داده شامل آدرس های مبدا، مقصد، شماره بسته و مکانیزم کشف خطا و... نیز هست.

مسیریابی هر بسته به صورت مستقل انجام می شود. ترتیب دریافت بسته ها در سمت گیرنده حفظ نمی شود.

Subject:
Year. Month. Date. ( )

مقایسه روش بسته ای نسبت به روش عددی

فرستنده های مختلف به طور همزمان می توانند از یک کانال استفاده کنند اگرچه سرعت پایین تری نسبت به سوئیچینگ مداری دارد و ترتیب بسته ها توسط گیرنده حفظ نمی شود.

دریافت

مقایسه روش بسته ای نسبت به روش پیغام

حجم محدود بسته ها فضای حافظه کمتری نیاز دارد - سرعت بسته ای از پیغام بالاتر است - در صورت وقوع خطا در یک بسته فقط قسمت کوچکی از اطلاعات از بین می رود.
در شبکه اینترنت (لایه شبکه) از روش سوئیچینگ بسته ای داده گرام استفاده می شود.

آدرس دهی در لایه شبکه:

لایه پیوند داده وظیفه انتقال داده بین دو کامپیوتر در یک شبکه را بر عهده دارند و لایه شبکه مسئول مسیریابی و انتقال داده بین دو کامپیوتر در مجموعه ای از شبکه ها را بر عهده دارد.

پروتکل هایی که لایه شبکه دارند از یک آدرس منطقی مخصوص فرد به نام آدرس IP استفاده می کنند.
برای هر کامپیوتر یا مسیریاب یک آدرس IP تعریف می شود.

پروتکل های لایه پیوند داده فقط در یک شبکه محلی قابل استفاده اند و صرفا برای شناسایی کامپیوترهای درون یک شبکه از آدرس MAC استفاده می شود. آدرس MAC به صورت مسطح است و آدرس های IP به صورت سلسله مراتبی.

✓ آدرس IP: یک عدد ۳۲ بیتی مخصوص به فرد است که به هر کامپیوتر یا مسیریاب نسبت داده می شود. این آدرس IP را به صورت یک عدد دسیمال (ده دهی) نقطه گذاری شده نشان می دهند. در این روش هر عدد باینری ۳۲ بیتی به چهار بایت ۸ بیتی مجزا تقسیم شده که توسط نقطه از هم جدا می شوند. هر بایت توسط یک عدد دسیمال نمایش داده می شود.

(دودویی (باینری)) (ده دهی (دسیمال))

Subject:

Year. Month. Date. ( )

۱۷۷

تبدیل آدرس IP به عدد دسیمال: آنهایی که ۱ دارند

۰×۲۷ + ۱×۲۶ + ۰×۲۵ + ۱×۲۴ + ۱×۲۳ + ۰×۲۲ + ۰×۲۱ + ۱×۲۰ = ۹۲

۹۲.۵۷.۱۷۷.۱۷۸

تبدیل آدرس دسیمال به باینری: ۱۲۸ . ۱۱ . ۳ . ۳۱

۱۱۱۱۱۱۱۱ . ۰۰۰۰۰۰۱۱ . ۰۰۰۰۱۰۱۱ . ۱۰۰۰۰۰۰۰

۱۲۸ | ۲
۲

کلاس های آدرس IP:

A ۰ – ۱۲۷

B ۱۲۸ – ۱۹۱

C ۱۹۲ – ۲۲۳

D ۲۲۴ – ۲۳۹

E ۲۴۰ – ۲۵۵

کلاس D برای بخش گروهی استفاده می شود و به همراه کلاس E رزرو شده اند و در شبکه اینترنت از آنها استفاده نمی شود.

کلاس های آدرس A، B و C دارای دو قسمت شماره شبکه و شماره میزبان به صورت زیر می باشد:

A | شماره شبکه | شماره میزبان |

B | شماره شبکه | شماره میزبان |

C | شماره شبکه | شماره میزبان |

قسمت شماره شبکه برای مشخص کردن شبکه ای است که کامپیوتر یا مسیریاب به آن متصل است و قسمت شماره میزبان نیز شماره ی آن دستگاه متصل به شبکه است.

کلاس A دارای شماره شبکه ۱ بایتی و شماره میزبان ۳ بایتی

کلاس B دارای " ۲ " ۲ "

لم ۰ تا ۲۵۴ میزبان را می توانند مشخص کنند

Subject:

Year. Month. Date. ( )

کلاس C داری شماره شبکه ۳ بایتی و شماره میزبان ۱ بایتی است.

* شرکتی با ۱۰۰ ایستگاه کاری نیاز به ۱۰۰ آدرس IP دارد. مهترین کلاس کلاس برای این شبکه مناسب است. کلاس C ، زیرا شماره میزبان آن یک بایتی است.