

شبکه های کامپیوتری برنامه های کاربردی شبکه - قسمت 1

سیامک سرمدی، وحید سلوک

1

فهرست مطالب

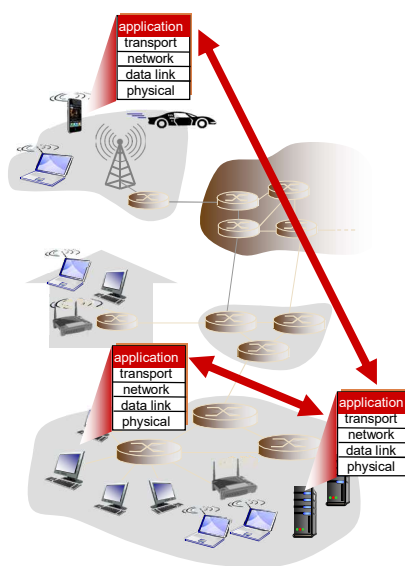
- معماری برنامه ها یا کاربردهای شبکه ای
- سرویس های لایه انتقال
- پروتکل های لایه کاربرد
- معماری وب و پروتکل HTTP
- پروتکل انتقال فایل (FTP)
- پست الکترونیکی
- SMTP, POP3, IMAP
- سرویس دهنده نام دامنه (DNS)
- برنامه های کاربردی P2P

2

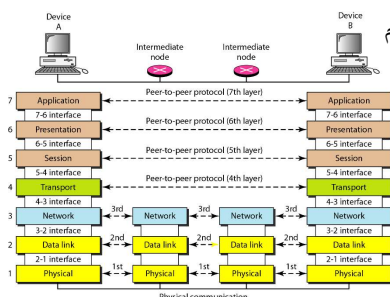
معماری برنامه ها یا کاربردهای شبکه ای

3

معماری برنامه ها یا کاربردهای شبکه ای (Network Applications)

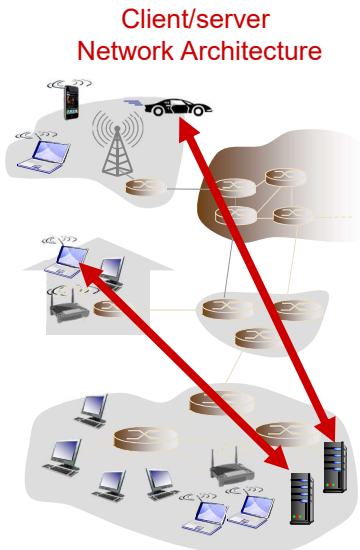


- هدف اصلی ایجاد ارتباطات شبکه، ارسال داده ها بین برنامه های کاربردی است. این برنامه ها:
 - معمولاً در سیستم های انتهایی اجرا میشوند.
 - در تجهیزات هسته شبکه که معمولاً در لایه کاربرد یا لایه 5 کار نمیکنند وجود ندارند.
 - اجزاء یک کاربرد شبکه از شبکه برای ارتباط استفاده می کنند.
- ارتباط برنامه سرویس دهنده وب (Web Server) و مرورگرهای وب (Web browser)
- ارتباط بین برنامه های p2p، مشابه، با هم



4

معماری کاربردهای شبکه ای - معماری سرویس دهنده-مشتری



**Client/server
Network Architecture**

سرویس دهنده (server):

- میزبان های همیشه روشن و فعال
- معمولاً دارای آدرس های اینترنتی ثابت (IP Address)
- در کاربردهای پرطرفدار یک سرور تنها قادر به ارائه سرویس نیست. تعدادی سرور مستقر در یک یا چند مرکز داده (data center) مورد استفاده قرار میگیرند.
- برنامه وب سرور که فایل های وب را به مرورگرها ارسال میکند
- برنامه های سرور FTP، Telnet، SSH، Email، Telegram

مشتری (client):

- قابلیت ارتباط با برنامه سرویس دهنده
- دارای آدرس های اینترنتی متغیر یا ثابت
- لزوماً همیشه فعال نیستند
- معمولاً مشتری ها با همدیگر ارتباط مستقیم برقرار نمیکنند
- برنامه های مرورگر وب، ترمینال SSH، کلاینت ایمیل و برنامه Telegram

5



peer-peer

معماری نظیر-به-نظیر (Peer-to-Peer, P2P)

- **عدم نیاز** به سرویس دهنده های همیشه فعال مرکزی
- میزبان ها بطور **مستقیم با هم** ارتباط برقرار میکنند (ارتباط نظیر به نظیر)
- میزبان های p2p معمولاً متعلق به **کاربران** (خانگی، شرکت ها و ...) هستند
- **درخواست** سرویس توسط یک نظیر (peer) از نظیر دیگر صورت میگیرد
- ارائه سرویس ها توسط همان نظیر دیگر
- لزوماً، همیشه فعال نیستند، و ممکن است مرتب آدرس های اینترنتی آنها تغییر کند.
- **مزایا:**
 - **مقیاس پذیری** با افزوده شدن نظیر های دیگر
 - **هزینه پایین** به علت عدم نیاز به زیرساخت و پهنای باند زیاد
- **معایب:** پیچیدگی پیاده سازی و مدیریت، تهدیدات امنیتی، مغایرت با سیاست ISP، قابلیت اطمینان و بازده آنها به علت توزیعی بودن ممکن است پایین باشند.

6

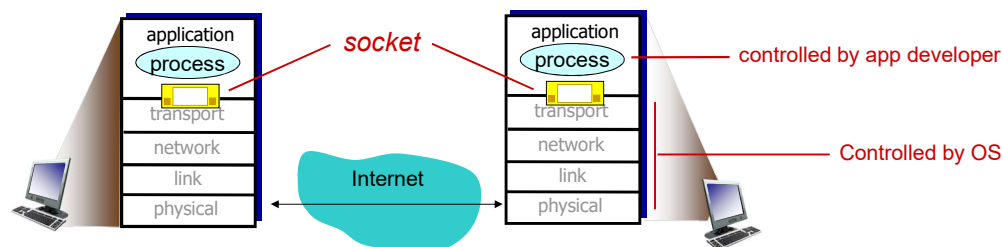
ارتباط پردازش ها (Process Communication)

- پردازش (process)
 - برنامه ی در حال اجرا در یک میزبان
- ارتباط بین پردازش ها
 - ارتباط بین دو پردازش در یک میزبان معمولا از طریق روال های سیستم عامل (پایپ، فایل، حافظه مشترک) انجام می شود
 - ارتباط بین دو پردازش از دو میزبان متفاوت از طریق تبادل پیام (message passing) انجام می شود.
- پردازش ها در معماری P2P
 - پردازش ها در ارتباط های مختلف باهم گاهی در نقش مشتری و گاهی در نقش سرویس دهنده عمل میکنند.
- پردازش ها در معماری سرویس دهنده-مشتری
 - پردازش مشتری (client process) به پردازش آغاز کننده ارتباط گفته میشود.
 - پردازش سرویس دهنده (server process) معمولا منتظر درخواست میماند.

7

ارتباط پردازش و شبکه

- پیام های بین پردازش ها از شبکه عبور می کند
- **سوکت (دروازه):** لایه نرم افزاری که پردازش ها برای ارسال و دریافت پیامهای شبکه از آن استفاده میکنند.
 - اگر پردازش ها را بعنوان خانه در نظر بگیریم، سوکت ها درهای خانه هستند.
 - واسط بین لایه کاربرد و لایه انتقال در میزبان (واسط برنامه نویسی کاربردی یا API)
- **موارد قابل کنترل توسط برنامه نویس در لایه کاربرد:** محدود به انتخاب پروتکل انتقال (tcp, udp)، اعمال تنظیمات محدود (مثلا حداکثر سائز بافر و حداکثر سائز هر سگمنت)



8

آدرس دهی پردازش ها

- برای ارسال پیام به میزبان مقصد به موارد زیر نیاز هست:
 - آدرس اینترنتی میزبان مقصد: آدرس IP میزبان مقصد
 - شناسه پردازش گیرنده در میزبان مقصد: شماره یا مشخصه ای به نام پورت که توسط برنامه نویس مشخص شده و یا بطور پویا توسط سیستم عامل اختصاص می یابد. آدرس نقطه سرویس هم نامیده میشود.
- آدرس های نقاط سرویس:
 - آدرس های استاندارد: به محدوده 1024 شماره اول که به سرویس های ثبت شده در سازمان IANA اختصاص میابند گفته میشود.
 - آدرس های توسعه: بقیه آدرس ها تا شماره 65535 که قابل استفاده برای سایر برنامه ها میباشند. معمولا محدوده 1024-49151 این رنج برای برنامه های سرور ثبت نشده در IANA و رنج 49152 به بالا توسط سیستم عامل بصورت پویا به client ها اختصاص میابد.

9

سرویس های لایه انتقال

10

سرویس های انتقال (در دسترس برای لایه کاربرد)

- **لایه انتقال:** قبلا ذکر شد که برنامه های کاربردی لایه کاربرد، از سوکت برای ارتباط با لایه انتقال (Transport) استفاده میکنند.
 - شبکه ها (شبکه اینترنت TCP/IP، شبکه IPX و ...) معمولاً بیش از یک پروتکل لایه انتقال را در اختیار لایه کاربرد قرار میدهند.
 - برای انتخاب از بین پروتکل های لایه انتقال، معمولاً به سرویس هایی که هرکدام ارائه میدهد توجه میکنیم.
 - انتخاب پروتکل لایه انتقال مانند انتخاب وسیله نقلیه اتومبیل، ترن و یا هواپیما برای حمل و نقل است. هرکدام از این وسایل نقلیه سرویس های متفاوتی را ارائه میکنند (بار سنگین تر، سرعت حمل، هزینه کمتر، امنیت بالاتر و ...)
 - سرویس هایی که پروتکل های لایه انتقال به برنامه های کاربردی ارائه میکنند در چهار بعد تقسیم میگردند:
- (1) انتقال مطمئن داده:** بعضی کاربردها نیاز به قابلیت اطمینان 100% در انتقال داده دارند (حتی 1 بیت داده نباید تغییر کند)
- کاربردهای انتقال فایل، ارسال اسناد وب و ... نیاز به انتقال مطمئن دارند
 - کاربردهایی که داده های صوتی و تصویری منتقل میکنند، تحمل پذیری بیشتری نسبت به حذف داده دارند (loss-tolerant)

11

سرویس های انتقال (در دسترس برای لایه کاربرد)

- (2) تامین گزردهی (throughput) مناسب:** مقدار گزردهی در دسترس شبکه برای یک برنامه، ممکن است در طول زمان تغییر کند. بعضی از پروتکل های انتقال میتوانند مقدار مشخصی از گزردهی بر حسب bit/s را ضمانت کنند.
- برخی برنامه ها به مقدار مشخصی از آستانه گزردهی حساس هستند (Bandwidth sensitive applications)
 - برنامه های چند رسانه ای
 - بعضی از برنامه ها میتوانند با هر اندازه از گزردهی به کار خود ادامه دهند (Elastic applications)
- (3) تاخیر زمانی مشخص:** بعضی از پروتکل های لایه انتقال میتوانند در مورد زمان دریافت پیام ضمانت بدهند.
- برخی برنامه ها به تاخیر پایین و مشخصی برای کارکرد بهتر نیاز دارند (مثلاً کمتر از 100 میلی ثانیه)
 - تلفن اینترنتی: تاخیر در ارسال صوت مکث های غیر عادی ایجاد میکند.
 - بازی های تعاملی: نیاز به زمان عکس العمل مناسب دارند
- (4) تامین امنیت:** بعضی از پروتکل های لایه انتقال سرویس هایی را در زمینه امنیت داده ها ارائه میکنند.
- محرمانگی (encryption)
 - جامعیت (integrity)
 - تایید هویت (authentication)

12

سرویس های انتقال ارائه شده توسط اینترنت

- شبکه اینترنت که بر اساس پروتکل TCP/IP کار میکند دو پروتکل لایه انتقال (Transport) را ارائه میکند که عبارتند از UDP و TCP
- هنگام طراحی یک برنامه کاربردی اولین تصمیمی که باید بگیرید، در مورد استفاده از یکی از این 2 پروتکل است.
- هر کدام از این 2 پروتکل سرویس ها و مشخصاتی را ارائه میکنند.
- جدول زیر نشان میدهد تعدادی از برنامه های کاربردی مرسوم چه نیازمندیهایی را برای انتقال دارند.

Application	Data Loss	Throughput	Time-Sensitive
File transfer/download	No loss	Elastic	No
E-mail	No loss	Elastic	No
Web documents	No loss	Elastic (few kbps)	No
Internet telephony/ Video conferencing	Loss-tolerant	Audio: few kbps–1Mbps Video: 10 kbps–5 Mbps	Yes: 100s of msec
Streaming stored audio/video	Loss-tolerant	Same as above	Yes: few seconds
Interactive games	Loss-tolerant	Few kbps–10 kbps	Yes: 100s of msec
Smartphone messaging	No loss	Elastic	Yes and no

13

سرویس های انتقال ارائه شده توسط اینترنت - TCP

- TCP: پروتکلی است که یک ارتباط "اتصال گرا" (connection oriented) و مطمئن (reliable) را ارائه میکند.
- سرویس اتصال گرا: قبل از ارسال داده ها، مشتری درخواست ارتباط را به مشتری ارسال میکند. پس از رد و بدل شدن چند پیام (دست دهی یا handshaking)، داده ها بین دو میزبان جریان میابند. پس از انجام دست دهی، گفته میشود که یک اتصال بین سوکت های دو پروسه یا پردازش برقرار شده. پس از پایان ارسال داده، برنامه کاربردی اتصال را با ارسال یک پیام به پایان میرساند.
- سرویس مطمئن: TCP قادر است بسته ها را بدون خطا و با ترتیب درست به پروسه مقصد برساند. اگر بسته ای خراب شده باشد، بسته مجددا دریافت میگردد و پس از مرتب کردن ترتیب بسته ها، داده (پیام) استخراج شده و تحویل برنامه کاربردی می گردد.
- کنترل ازدحام: TCP همچنین از یک مکانیزم کنترل ازدحام نیز بهره میبرد. در صورتی که گیرنده بر اثر ازدحام مسیر مابین فرستنده و گیرنده نتواند بسته های ارسالی را دریافت کند، این مکانیزم سرعت ارسال بسته ها توسط مبدأ را کاهش میدهد.

14

سرویس های انتقال ارائه شده توسط اینترنت - UDP

- UDP: پروتکلی است که **سبک** بوده، **بدون اتصال** است و **حداقل سرویس** را ارائه میکند.
 - دست دهی اولیه و برقراری اتصال در این پروتکل وجود ندارد.
 - پروتکلی **غیر قابل اطمینان** است و **تحویل** بسته ها و **ترتیب** رساندن آنها تضمین نشده است. ممکن است بسته ها به مقصد نرسند و یا ترتیب بسته های دریافت شده با سمت فرستنده متفاوت باشد.
 - UDP مکانیزمی برای **کنترل ازدحام ندارد** و در صورت نیاز **تنظیم سرعت** ارسال باید در **لایه کاربرد** انجام شود.
 - **فرستنده** میتواند با **هر سرعت** مورد نظری ارسال کند ولی مقدار داده دریافتی به **ظرفیت و ازدحام لینک** های بین فرستنده و گیرنده بستگی دارد.

15

برنامه های کاربردی اینترنت

- در جدول زیر **پروتکل انتقال** مورد استفاده توسط بعضی از کاربردهای اینترنتی لیست شده است.

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP [RFC 5321]	TCP
Remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
File transfer	FTP [RFC 959]	TCP
Streaming multimedia	HTTP (e.g., YouTube)	TCP
Internet telephony	SIP [RFC 3261], RTP [RFC 3550], or proprietary (e.g., Skype)	UDP or TCP

16

سرویس های انتقال ارائه نشده توسط اینترنت

□ **تضمین گزدهی و تضمین های زمانی :** در پروتکل های TCP و UDP، این سرویس ها ارائه نشده است.

□ **سوال:** با توجه به ارائه نشدن سرویس های فوق کاربردهای تلفن اینترنتی چطور قابل اجرا هستند؟

□ **طراحی مناسب لایه کاربرد:** این کاربردها بر روی اینترنت توانایی ارائه سرویس قابل قبولی را دارند. علت این است که این کاربردها به نحوی طراحی شده اند که بتوانند تا حد ممکن با این کمبود های شبکه اینترنت و عدم ارائه ضمانت در این دو زمینه کنار بیایند.

■ این کاربردها مانند skype معمولاً میتوانند با مقداری تلفات بسته (loss) کنار بیایند.

■ در این کاربردها باید مقدار تاخیر کم باشد، بنابراین در کاربردهای صوتی معمولاً از پروتکل UDP استفاده میشود که بار اضافه بسته ها، تبادل پیامهای اضافه و نیاز به ایجاد اتصال از بین بروند.

■ با توجه به اینکه ورود بسته های UDP در بعضی شبکه ها مسدود شده، پروتکل TCP نیز معمولاً بعنوان جایگزین برای پروتکل UDP ممکن است مورد استفاده قرار بگیرد.

17

سرویس های انتقال ارائه نشده توسط اینترنت

□ **امنیت:** پروتکل های TCP و UDP مکانیزم خاصی را برای برقراری امنیت ارائه نمیکنند.

□ **اضافات SSL:** جامعه اینترنت اضافاتی را برای پروتکل TCP ایجاد کرده است که با نام لایه سوکت امن (Secure Socket Layer) یا SSL شناخته میشود.

■ این اضافات لایه جدیدی نیستند و معمولاً با کمک کتابخانه های برنامه نویسی در لایه کاربرد پیاده سازی میشوند. برای ارتباط از طریق SSL کتابخانه های سوکت مجزائی در دسترس هستند که بجای کتابخانه سوکت معمولی مورد استفاده قرار میگیرند و ارتباط امن "پروسه به پروسه" را ممکن میکنند.

■ در صورت استفاده از این اضافات، با ارائه سرویس های جدید، امنیت ارتباط "پروسه به پروسه" برقرار میگردد.

■ این سرویس ها عبارتند از رمز گذاری (encryption)، جامعیت داده (integrity) و اعتبار سنجی (authentication)

18

پروتکل های لایه کاربرد

19

پروتکل های لایه کاربرد

- همانطور که قبلا ذکر شد برنامه های کاربردی با تبادل پیام ارتباط برقرار میکنند. ساختار این پیام ها توسط پروتکل های لایه کاربرد تعریف میگردند. مواردی که توسط این پروتکل ها تعریف میگردند عبارتند از:
 - انواع پیامها: انواع پیامهایی که رد و بدل میشوند (پیام درخواست، پیام پاسخ، پیام تایید و ...)
 - قالب دستوری پیام ها (syntax): شامل فیلدهای تشکیل دهنده پیام های مختلف و اینکه مقدار این فیلدها چطور مشخص میشود.
 - مفهوم پیام و فیلدهای آن (semantic): شامل معنای اطلاعات قرارگرفته در فیلدها
 - قواعد: که مشخص میکنند چه زمانی و چطور ، پروسه ها پیامها را ارسال میکنند و یا به آنها پاسخ میدهند.
- بعضی از پروتکل های لایه کاربرد به شکل RFC در اختیار عموم قرار داده شده اند و بعضی دیگر که مربوط به برنامه های کاربردی ساخته شده توسط شرکت ها هستند (مانند Skype) در اختیار عموم قرار نگرفته اند.
 - از جمله پروتکل های باز میتوان به پروتکل HTTP اشاره کرد که در RFC های 1945 و 2616 تشریح شده است.
 - یکی دیگر از پروتکل های پر مصرف پروتکل SMTP (Simple Mail Transfer Protocol) است که در RFC821 و RFC5321 تعریف شده است.

20

پروتکل های لایه کاربرد

- یک مورد مهم که باید به آن توجه کرد تفاوت بین پروتکل لایه کاربرد (Network-layer Protocol) و برنامه کاربردی شبکه (Network Application) است.
- کاربرد وب (Web): از چند جزء تشکیل شده است
 - استانداردهای مربوط به اسناد تبادل شده روی وب: فرمت فایل های رد و بدل شده بر روی وب را تشریح میکنند. از جمله استاندارد HTML نسخه 5.0 فرمت و ساختار فایل های html را تعریف میکند.
 - مرورگر (browser): نرم افزاری است که به عنوان مشتری (client) عمل کرده، صفحات وب را دریافت کرده و محتوای آنها را بصورت گرافیکی نمایش میدهد.
 - سرور وب (web server): صفحات و سایر اسناد وب را میزبانی کرده و مطابق با درخواست های مرورگر، آنها را به مرورگر ارسال میکند.
 - پروتکل HTTP: فرمت و مشخصات پیامهای رد و بدل شده بین مرورگر و وب سرور را تعریف مینماید.
- کاربرد ایمیل: بصورت مشابه کاربرد ایمیل شامل اجزاء مختلفی است که شامل نرم افزار مشتری (mail client)، نرم افزار سرور ایمیل (email server)، استاندارد های مربوط به ساختار فایل های غیر متنی تبادل شده (MIME)، و پروتکل ارتباطی که ساختار درخواست و پاسخ ها را مشخص میکند (SMTP Protocol) می باشد.

21

وب و پروتکل HTTP



Sir Tim Berners-Lee

- درسالهای اول دهه 1990 اینترنت بیشتر توسط محققان، اساتید و دانشجویان برای اتصال به میزبانها از راه دور (با نرم افزار ترمینال Telnet)، ارسال و دریافت خبر (با newsgroup ها) و ارسال و دریافت ایمیل مورد استفاده قرار میگرفت و تقریباً در محیط های بیرون دانشگاه ناشناخته بود.
- در سال 1994 یک کاربرد جدید اینترنتی به نام وب جهانی (world wide web) که توسط تیم برنرز لی (Tim Berners-Lee) ابداع شده بود ارائه شد و نحوه استفاده از اینترنت در محیط کار و بیرون از آن و همچنین گستردگی استفاده از اینترنت را تغییر داد.

22

وب و پروتکل HTTP

- وب باعث شد که اینترنت به پر استفاده ترین شبکه از میان شبکه های موجود تبدیل شود. برتری های وب نسبت به سایر سرویس های موجود:
 - کاربران قادرند فقط به اطلاعات مورد نیاز خود و در زمان مورد نظر خود (On demand) دسترسی پیدا کنند.
 - کاربران قادر به انتشار مطالب و محتوای تولید شده توسط خود هستند.
 - موتورهای جستجو، امکان یافتن مطالب مورد نظر کاربر را فراهم مینمایند.
 - فرم ها و اسکریپت ها (جاوا اسکریپت) امکان تعامل کاربر با صفحه وب را فراهم میکنند.
 - امکان ارائه سرویس های دیگر مانند ایمیل، کنفرانس ویدئویی یا صوتی و ... از طریق وب و بصورت ساده تر ممکن است.
- استانداردها و پروتکل های مختلفی در کاربرد وب مورد استفاده قرار میگیرند. از جمله:
 - پروتکل HTTP: ساختار پیامهای رد و بدل شده
 - استاندارد HTML: ساختار فایل صفحات متنی رد و بدل شده

23

مرور کلی پروتکل HTTP

- پروتکل HTTP (HyperText Transfer Protocol): پروتکل کاربرد وب است که به منظور انتقال ابرمتن (فایل های html) و متعلقاتشان ایجاد شده و جزئیات آن در RFC1945 و RFC2616 تعریف شده است.
- پروتکل HTTP در دو برنامه مرورگر (web browser) و سرور وب (web server) پیاده سازی شده است.
- دو برنامه فوق با تبادل پیامهایی که ساختار آنها مطابق پروتکل HTTP ساخته شده اند باهم صحبت میکنند.
- مرورگر وب معمولاً با این پیامها، اشیاء یا فایل هایی را از سرور درخواست میکند و سرور فایل مورد نظر را به مرورگر ارسال میکند.
- صفحات وب: معمولاً از یک فایل با ساختار html تشکیل میگردند که داخل آن تعدادی فایل تصویر، اسکریپت، فونت و منابع دیگر که اشیاء وب نامیده میشوند، مورد رجوع قرار گرفته اند.
- یک مستند (document) یا شیئی وب (Object) یک فایل، مانند فایل html، تصویر jpeg و یا فایل صوتی است.
- URL: هر شیء توسط یک آدرس منحصر به فرد که میزبان سرویس دهنده و محل فایل در میزبان سرویس دهنده را مشخص میکند، مشخص میگردد.

www.uut.ac.ir/somefolder/someimagefile.jpg



24

پروتکل HTTP



این پروتکل مبتنی بر معماری سرویس دهنده-مشتری طراحی شده

مشتری یا مرورگر (client/browser): برنامه ای برای درخواست، دریافت اشیاء وب (با کمک پروتکل HTTP) و اجرا و نمایش آنها میباشد. این برنامه ابتدا برای صفحه وب مورد نظر و سپس اشیاء رجوع شده در داخل آن با پروتکل انتقال TCP به سرور درخواست میفرستند.

سرویس دهنده (server): سرویس دهنده وب درخواست ها را میگیرد، منبع مورد نظر را یافته یا با کمک اسکریپت ی ا برنامه های سمت سرور تولید میکند و به مرورگر میفرستد (با پیامهای HTTP)

مراحل اتصال و تبادل اشیاء:

- مرورگر با ایجاد یک سوکت روی پورت تصادفی، به آدرس پردازش شماره 80 در سرویس دهنده درخواست یک اتصال TCP میفرستد.
- اتصال TCP توسط سرویس دهنده پذیرفته میشود.
- پیام های HTTP بین سرویس دهنده (HTTP Server) و مشتری (مرورگر) تبادل میشوند.
- اتصال TCP خاتمه میابد.

25

پروتکل HTTP

وب سرور:

- همیشه روشن است
- معمولاً یک آدرس ثابت IP دارد
- ممکن است به میلیونها مرورگر پاسخ بدهد.

پروتکل بدون حالت (stateless): از آنجائیکه وب سرور اطلاعاتی را در مورد کلاینت (مرورگر) (state) نگه نمیدارد، پروتکل HTTP یک پروتکل بدون حالت نامیده میشود.

26

پروتکل HTTP - اتصال پایا و غیر پایا

□ اتصال HTTP غیرپایا (non-persistent):

- ارسال هر درخواست (برای دریافت یک شیء) و دریافت پاسخ آن با برقراری یک **اتصال مجزای TCP** انجام شده و اتصال قطع میشود.
- برای دریافت اشیاء **دیگر** (مثلا تصاویر روی یک صفحه html) نیاز به برقراری **اتصال های TCP دیگری** است.
- پس از هر انتقال اتصال قبلی بسته شده و اتصال جدیدی برای انتقال بعدی ایجاد می شود.

□ اتصال HTTP پایا (persistent):

- از نسخه 1.1 پروتکل HTTP به بعد معرفی گردید.
- کلیه درخواست و پاسخ های HTTP **روی یک اتصال TCP** که **باز نگه داشته** می شود، انجام میگردد.
- HTTP بطور پیشفرض از **اتصال پایا** استفاده میکند ولی سرور و کلاینت های HTTP را میتوان طوری **تنظیم** کرد که از اتصال های **غیر پایا** استفاده کنند.

27

پروتکل HTTP – مراحل اتصال غیر پایا

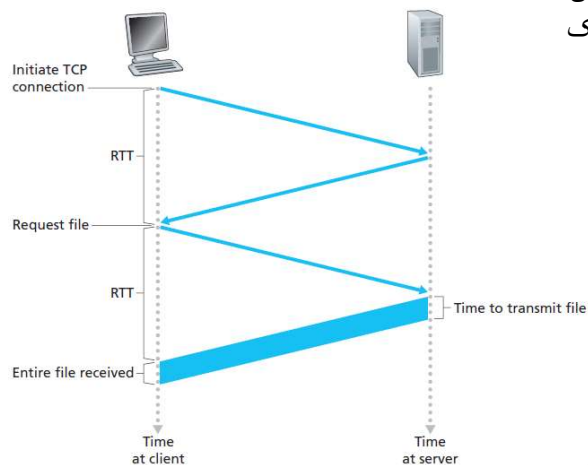
□ مراحل اتصال HTTP غیرپایا (non-persistent):

1. کلاینت یک ارتباط را به آدرس URL سرور و بر روی **پورت 80 آغاز** میکند. **شماره پورت** کلاینت بصورت تصادفی توسط سیستم عامل تعیین شده است.
2. کلاینت پیام **درخواست** خود، که شامل آدرس مسیر مثلا `/departments/index.html` است را به سرور میفرستد
3. سرور درخواست را **دریافت** میکند و **شیء** مورد نظر (که آدرس مسیرش دریافت شده) را از دیسک یا حافظه **استخراج** میکند، آنرا در یک پیام کپسوله میکند و به کلاینت میفرستد.
4. سرور به TCP میگوید که **ارتباط را قطع** کند. TCP تنها پس از اطمینان از دریافت صحیح فایل توسط کلاینت، ارتباط را قطع میکند.
5. کلاینت پاسخ را **دریافت** کرده و ارتباط TCP قطع میگردد (از سمت سرور). شیء کپسوله شده در پیام استخراج میشود. کلاینت متوجه میشود که فرمت فایل html است و **10 تصویر JPG** در آن استفاده شده اند.
6. مراحل 1 تا 4 برای هرکدام از تصاویر تکرار میشود (مجموعاً 11 ارتباط که **ممکن** است به **موازات** انجام شوند که دریافت اطلاعات صفحه را بسیار سریعتر می کند).

- نکته: 2 مرورگر متفاوت ممکن است یک فایل را بطور **متفاوتی تفسیر** کرده و نمایش بدهند. این مسئله به **پیاده سازی فرمت HTML** ربط دارد و به پیاده سازی پروتکل ارتباطی HTTP ربط ندارد.

28

تحلیل زمانی اتصال غیرپایا – Three-way-handshake



□ **زمان رفت و برگشت (Round Trip Time):** زمان ارسال یک بسته کوچک به سرویس دهنده و دریافت یک بسته کوچک پاسخ (حجم کم)

□ تاخیر پخش

□ تاخیر صف

□ تاخیر پردازش

□ **فرض کنید کاربر بر روی یک لینک کلیک میکند:**

□ یک RTT برای برقراری اتصال TCP

□ یک RTT درخواست HTTP و دریافت چند بایت اول پاسخ

□ زمان انتقال فایل

□ **زمان کل پاسخ پایا:** $2RTT + \text{file transmission time}$

□ **زمان کل پاسخ غیر پایا:** نیاز به 2 زمان RTT برای هر شیء

29

ساختار پیام های درخواست HTTP

□ در RFC های 1945، 2616 و 7540 تشریح شده است.

□ با کد ASCII

□ یک پیام معمول HTTP از دستورات زیر تشکیل شده و در انتهای هر سطر علامت های `\n` و `\r` قرار دارد.

```

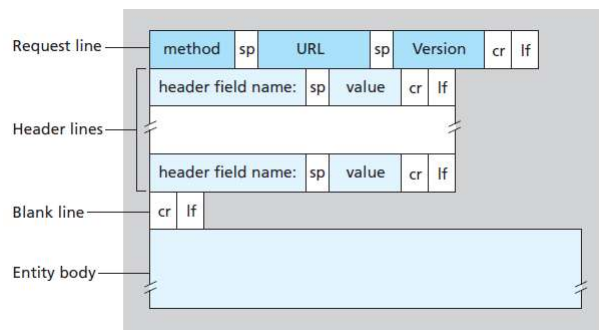
request line
(GET, POST,
HEAD commands)
GET /index.html HTTP/1.1\r\n
header
lines
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
Entity Body
carriage return character
line-feed character
میتواند close باشد.

```

30

ساختار پیام های درخواست HTTP

- **Header Host:** برای تشخیص سایت مورد نظر توسط پراکسی سرور و همچنین ویرچوال هاست
- **Header Connection:** به سرور اطلاع میدهد که آیا باید اتصال را بعد از ارسال فایل ببندد یا برای درخواست های بعدی باز نگه دارد
- **Header Agent:** مفید است زیرا سرور میتواند نسخه های مختلفی را برای مرورگرهای مختلف ارسال کند.
- **قسمت Entity body:** در دستور Get خالی ولی در دستور Put و Post برای ارسال اطلاعات فرم و فایل استفاده می شود.



31

متدهای درخواست

- **متد GET**
 - نام دیگر: متد URL
 - برای درخواست یک شیء استفاده میشود
 - اطلاعات ارسالی (پارامترها یا اطلاعات فرم) بر روی URL قرار میگیرد
- **متد POST**
 - قابل استفاده برای فرم های قابل پر کردن
 - اطلاعات تکمیل شده در متن پیام و بدون دیده شدن روی URL از مرورگر به سرور منتقل می شود
 - دارای قسمت بدنه موجودیت (entity body)
- **متدهای دیگر**
 - **PUT:** ارسال یک شیء به سرور دهنده
 - **DELETE:** حذف یک شیء از سرور دهنده
 - **HEAD:** درخواست ارسال پاسخ خالی (برای خطایابی ارتباط)

33

ساختار پیام های پاسخ HTTP

□ شامل خط وضعیت، خطوط هدر، و بدنه موجودیت

□ قسمت اصلی بدنه موجودیت (داده) می باشد که معمولاً محتوی فایل یا شیئی ارسالی است

Status line:

- protocol
- status code
- status msg.

header lines

Entity body:

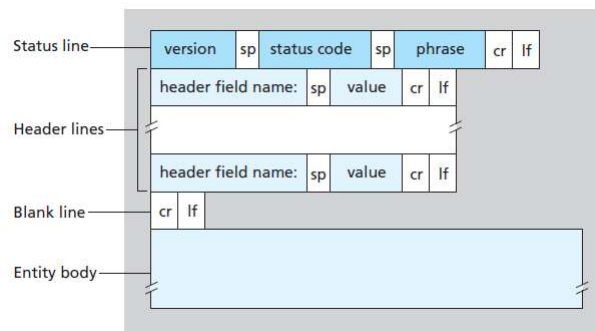
data, e.g.,
requested
HTML file

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02
GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-
1\r\n
\r\n
data data data data data ...
```

34

ساختار پیام های پاسخ HTTP

□ ساختار کلی پیام پاسخ



35

ساختار پیام های پاسخ HTTP

- خط وضعیت شامل نسخه پروتکل مورد استفاده، کد وضعیت پاسخ، و پیام مربوط به وضعیت پاسخ است.
- Date: زمان استخراج و ارسال فایل یا شیئی
- Server: نسخه و نام نرم افزار سرور وب و گاهی سیستم عامل و نام و نسخه زبان برنامه نویسی (مثلا PHP)
- Last-modified: آخرین تغییر شیئی مورد نظر (به درد سرورهای Cache و Proxy میخورد)
- E-Tag: نسخه شیء یا فایل مورد درخواست را میدهد. اجازه میدهد فایل ها با دقت بیشتری Cache شوند.
- Accept-Ranges: پشتیبانی از ارسال قسمت هایی از فایل (واحد مورد استفاده برای ذکر محدوده)
- Content-Length: طول داده ارسالی (بایت)
- Keep-Alive: زمان timeout و مدت زمان Keep-Alive تنظیم شده روی سرور را اعلام میکند.
- Connection: سرور اعلام میکند که با روش پایا یا غیر پایا کار میکند (Keep-Alive, Close)
- Content-Type: نوع محتوای ارسالی را مشخص میکند (این فیلد تعیین کننده است، نه پسوند فایل)

36

کد های وضعیت پیام پاسخ

- 200 OK موفقیت آمیز بودن درخواست، شیء در ادامه پاسخ قرار میگیرد
- 301 شیء درخواستی بطور دائمی از سرور سرور حذف شده
- 400 عدم تشخیص درخواست توسط سرور
- 404 سند درخواستی در سرور پیدا نشد
- 505 عدم پشتیبانی نسخه مورد استفاده پروتکل HTTP توسط سرور

نوع کد	مفهوم	توصیف
1yy	اطلاعاتی	شامل اطلاعات عمومی، بدون اطلاعات نتیجه
2yy	موفق	متمم توسط سرور دریافت و پذیرش شد
3yy	تغییر مسیر	نیاز به عملیات بیشتر قبل از پذیرش درخواست
4yy	خطای کاربر	درخواست نامعتبر، عدم دریافت کامل یا دارای خطای دستوری
5yy	خطای سرور	درخواست معتبر، عدم پذیرش بدلیل مشکلات سرور

37

کدهای وضعیت (موفق)

Code	Phrase	Description
Informational		
100	Continue	The initial part of the request has been received and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
Success		
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

38

کدهای وضعیت (ناموفق)

Code	Phrase	Description
Redirection		
301	Multiple choices	The requested URL refers to more than one resource.
302	Moved permanently	The requested URL is no longer used by the server.
304	Moved temporarily	The requested URL has moved temporarily.
Client Error		
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
Server Error		
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable, but may be requested in the future.

□ در مورد وضعیت های 302 و 303، فیلد Location در هدر به محل فایل اشاره میکند.

39

کوکی (Cookie)

□ برای ردیابی کاربران توسط سرویس دهنده (به منظور شناسایی کاربر) مورد استفاده قرار میگیرد:

Session یا نشست:

- یک کوکی (فایل) کوچک شامل شماره مشخصه مشتری در مرورگر ذخیره میشود و هر بار به سرور ارسال میشود.
- اطلاعات اضافه مربوط به کاربر یا نشست در سمت سرور ذخیره میشود.
- با بستن مرورگر حذف میگردد.
- امنیت بالاتری دارد.

Cookie:

- یک کوکی (فایل) مفصل تر شامل شماره مشخصه اتصال و اطلاعات دیگر کاربر در مرورگر ذخیره میشود و هر بار به سرور ارسال میشود.
- اطلاعات اضافه معمولاً روی مرورگر ذخیره میگردد (طبیعتاً در دیتابیس سمت سرور نیز اطلاعات کاملتر مربوط به کاربر موجود است)
- بطور دراز مدت قابل استفاده است.
- امنیت پایین تری دارد.
- معمولاً اطلاعات ارسالی ذخیره شده در کوکی را قبل از ارسال به مرورگر در سمت سرور رمز میکنند.

موارد استفاده:

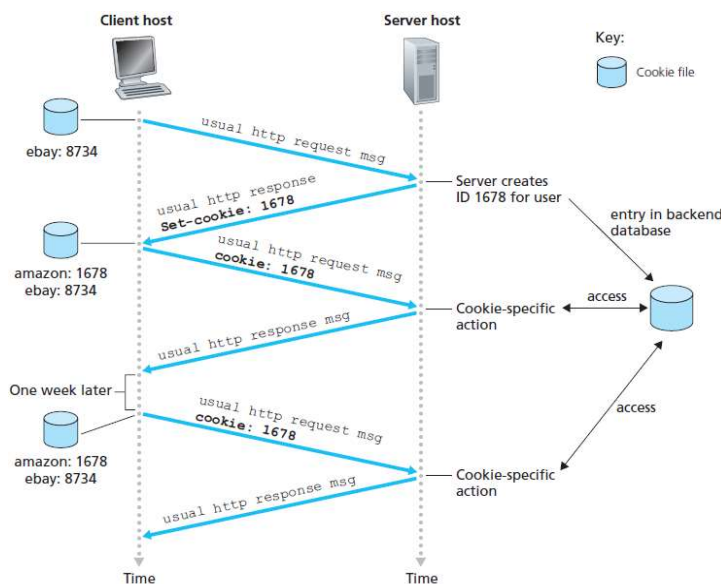
- احراز هویت
- سبد خرید
- پیشنهادات
- وضعیت نشست کاربر

40

کوکی (Cookie)

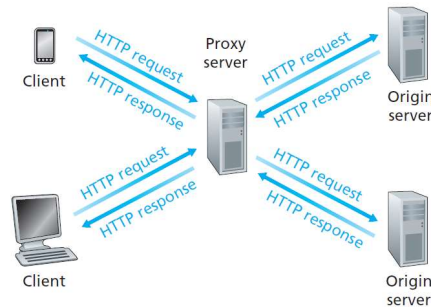
□ فزآیند استفاده از کوکی (session و cookie)

□ نکته: حساسیت نسبت به از بین رفتن حریم شخصی



41

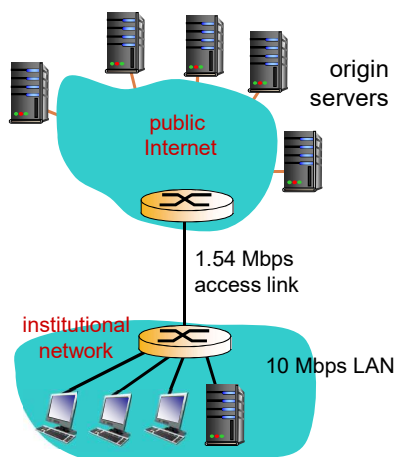
حافظه نهان وب (Web Cache)



- معروف به سرویس دهنده پراکسی (Proxy Server)
- واسطه سرویس دهنده وب برای پاسخ به درخواست ها بوده
- اتصال TCP توسط مرورگر به حافظه نهان انجام میگیرد و درخواست ها نیز ابتدا به پراکسی میروند
- در صورت وجود شیء درخواستی در پراکسی ← شیء به مشتری ارسال میشود.
- در صورت عدم وجود شیء در پراکسی یا قدیمی بودن نسخه موجود ← درخواست توسط پراکسی به میزبان وب ارسال شده، جواب دریافت و ذخیره میشود و سپس یک نسخه به مشتری ارسال میشود.
- هم در نقش سرور و هم در نقش کلاینت ظاهر میشود.
- معمولاً توسط ISP ها یا سازمانهای بزرگ بکار گرفته میشود.
- امکان استفاده با تنظیم مرورگر و یا نصب بصورت شفاف (transparent) وجود دارد.
- زمان پاسخ و مصرف پهنای باند و در نتیجه هزینه را شدیداً پایین میآورد.

42

حافظه نهان وب (Web Cache) – مثال 1: بدون حافظه ی نهان



assumptions:

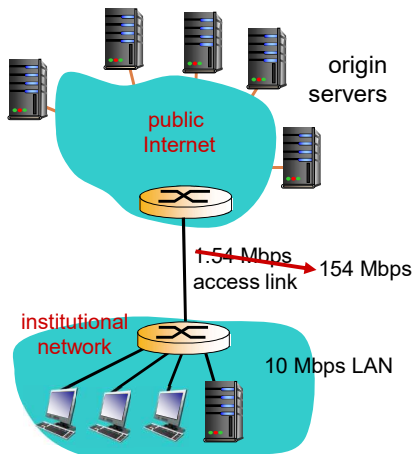
- ❖ avg object size: 100K bits
- ❖ avg request rate from browsers to origin servers: 15/sec
- ❖ avg data rate to browsers: 1.50 Mbps
- ❖ RTT from institutional router to any origin server: 2 sec
- ❖ access link rate: 1.54 Mbps

consequences:

- ❖ LAN utilization: 15%
- ❖ access link utilization = 99% *problem!*
- ❖ total delay = Internet delay + Transmission delay + LAN delay
= 2 sec + (for 100k file: 1s, for a 20MBfile: 2m) + msec

43

حافظه نهان وب (Web Cache) – مثال 1: بدون حافظه ی نهان، ارتقاء لینک دسترسی



assumptions:

- ❖ avg object size: 100K bits
- ❖ avg request rate from browsers to origin servers: 15/sec
- ❖ avg data rate to browsers: 1.50 Mbps
- ❖ RTT from institutional router to any origin server: 2 sec
- ❖ access link rate: 1.54 Mbps

consequences:

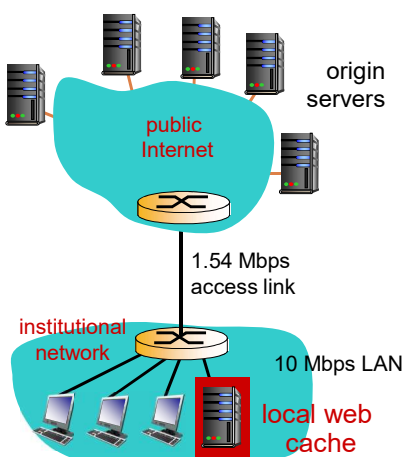
- ❖ LAN utilization: 15%
- ❖ access link utilization = 99%
- ❖ total delay = Internet delay + Transmission delay + LAN delay = 2 sec + minutes + msec

Seconds (for small file msec, for a 20MB file: 1.3 s)

Cost: increased access link speed (not cheap!)

44

حافظه نهان وب (Web Cache) – مثال 1: با حافظه ی نهان



assumptions:

- ❖ avg object size: 100K bits
- ❖ avg request rate from browsers to origin servers: 15/sec
- ❖ avg data rate to browsers: 1.50 Mbps
- ❖ RTT from institutional router to any origin server: 2 sec
- ❖ access link rate: 1.54 Mbps

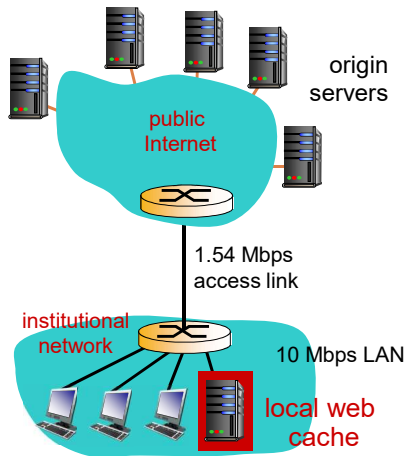
consequences:

- ❖ LAN utilization: 15%
- ❖ access link utilization = 58%
- ❖ total delay = Internet delay + Transmission delay + LAN delay = ~ 1.2 secs (for a small file)

Cost: web cache (cheap!)

45

حافظه نهان وب (Web Cache) – مثال 1: با حافظه ی نهان



□ نحوه محاسبه نتایج صفحه قبل:

suppose cache hit rate is 0.4
40% requests satisfied at cache,
60% requests satisfied at origin

Access link utilization:

- ❖ 60% of requests use access link
- ❖ data rate to browsers over access link = $0.6 * 1.50 \text{ Mbps} = .9 \text{ Mbps}$
- ❖ utilization = $0.9 / 1.54 = .58$

Total delay:

- ❖ = $0.6 * (\text{delay from origin servers}) + 0.4 * (\text{delay when satisfied at cache})$
- ❖ = $0.6 (2.01) + 0.4 (\sim \text{msecs for small file})$
- = ~ 1.2 secs
- ❖ Less than with 154 Mbps link (and cheaper too!)

Cost: web cache (cheap!)

46

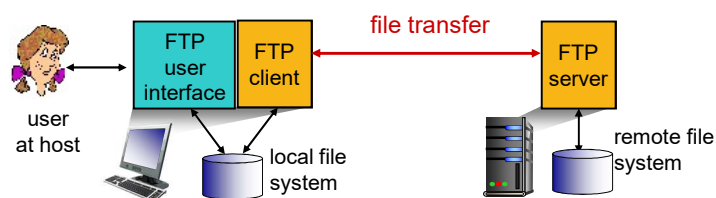
پروتکل انتقال فایل (FTP)

- پروتکلی برای انتقال فایل با قابلیت اطمینان و کارایی بالا
- امکانات ارائه شده توسط FTP
 - فهرست گیری از فایل های موجود روی سرور
 - حذف، تغییر نام و جابجایی فایل
 - ایجاد یا حذف شاخه
 - انتقال فایل از سرور به مشتری
 - انتقال فایل از مشتری به سرور
- استفاده از دو کانال مجزا برای انتقال داده و فرمان
- **کانال داده:** انتقال داده روی پورت 20
- **کانال فرمان:** مبادله فرامین روی پورت 21

47

فرآیند اتصال

- ارسال درخواست به سرویس دهنده روی پورت 21
- تایید هویت مشتری، مرور فایل ها و ارسال فرمان روی کانال کنترل
- دریافت فرمان انتقال فایل توسط سرویس دهنده
- برقراری اتصال دوم روی پورت 20
- خاتمه اتصال، باز کردن اتصال دوم برای انتقال فایل بعدی توسط سرویس دهنده
- نگهداری وضعیت توسط سرویس دهنده
- شاخه جاری، احراز هویت



48

فرمان های FTP

کدهای پاسخ ساده:

- 331, username OK
 - Password required
- 125 data connection already open; transfer starting
- 425 can't open data connection
- 452 error writing file

فرمان های ساده:

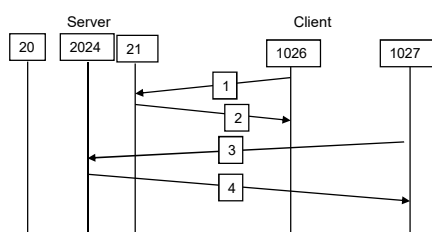
- USER username
- PASS password
- LIST
 - List current directory content
- RETR filename
- STOR file
 - puts file onto remote host

49

روش های ایجاد نشست

روش غیرفعال (Passive)

- ایجاد دو سوکت با شماره پورتهای تصادفی مشتری
- اتصال یکی از پورتهای با پورت 21 سرور
- درخواست نشست غیرفعال توسط مشتری
- ایجاد یک سوکت با شماره پورت تصادفی در سرور و اعلام آن به مشتری
- اتصال سوکت دوم مشتری با پورت اعلام شده توسط سرور و شروع نشست



روش معمولی (Normal/Active)

- ایجاد دو سوکت با شماره پورت تصادفی در مشتری
- اتصال یکی از پورتهای به پورت 21 سرور، اعلام آمادگی سرور برای تفسیر فرامین
- اعلام شماره پورت سوکت دوم به سرور و شروع به listen کردن روی آن توسط مشتری
- درخواست اتصال توسط سرور به پورت اعلام شده
- تصدیق درخواست توسط مشتری و شروع نشست

