دانشکده‌ی مهندسی کامپیوتر

شبکه‌های کامپیوتری مدرس: کامبیز میزانیان

تمرین سری اول (مقدمه و لایه‌ی کاربرد)

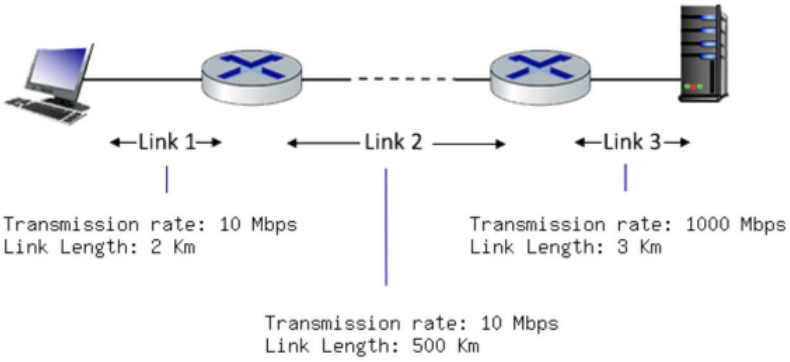
گردآورندگان: نیما نجفی - کیان بهادری - امیرحسین براتی - مهراد میلانلو

مهلت ارسال: ۱۸ فروردین ۱۴۰۳، ساعت ۲۳:۵۹

نکات تمرین:

- جواب سوالات را حتما به‌صورت **تایپ شده** و یا به‌صورت دستی و کاملا خوانا تحویل دهید.
- برای پاسخ به سوالات بخش آخر باید ابتدا نرم افزار `wireshark` را دانلود و نصب کنید.
- برای پاسخ به سوالات بخش آخر، در تمام مراحل اسکرین‌شات قسمتی از نرم‌افزار که جواب را از آن پیدا کرده‌اید هم بگذارید. اسکرین‌شات شما باید کل صفحه‌ی کامپیوترتان را نشان دهد.
- از تقلب و کپی کردن جواب دیگران جدا خودداری کنید! در صورت پیدا شدن هرگونه شباهت نامتعارف، مطابق با سیاست آیین‌نامه‌ی دانشکده عمل خواهد شد.
- سوالات و ابهامات خود را در پست مربوط به این تمرین در کوئرا مطرح کنید.

سؤال ۱ با توجه به شکل زیر به سوالات زیر پاسخ دهید. سرعت انتشار سیگنال روی لینک‌ها را $2 \times 10^8 m/s$ است.



(آ) با فرض ناچیز بودن زمان پردازش و زمان معطلی در صف، یک بسته $1500B$ بعد از چند ثانیه از *client* به *server* می‌رسد؟

(ب) اگر بخواهیم فایلی به طول $15000B$ را به بسته‌های $1500B$ تقسیم کرده و همه را پشت سر هم از *client* به *server* ارسال کنیم، با فرض ناچیز بودن زمان پردازش، کل فایل در چه مدت زمانی به سرور می‌رسد؟

(ج) حداکثر گذردهی *(Throughput)* در حالت ب چقدر است؟ چرا؟

(د) اگر بخواهیم زمان رسیدن فایل در حالت ب را به زیر $5ms$ برسانیم، چه راه‌حلی پیشنهاد می‌کنید؟

سؤال ۲ *HTTP* یک پروتکل بدون حالت *(state-less)* است. با این حال در طول روز با سایت‌هایی روبه‌رو می‌شویم که حالت هر *client* را به گونه‌ای حفظ می‌کنند؛ به طور مثال با یک بار *login* دیگر نیاز به ورود دوباره نیست، یا سایت‌های فروشگاهی‌ای که بدون *login* کردن هم سبد خرید شما را حتی بعد از *refresh* نگه می‌دارند. توضیح دهید این سایت‌ها از چه مکانیزمی برای *state-full* کردن این پروتکل استفاده می‌کنند.

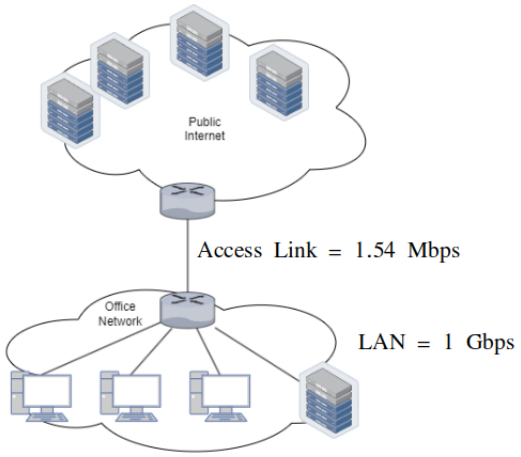
سؤال ۳ یک کاربر میخواهد از یک صفحه وب که شامل یک فایل *HTML* و 10 آبجکت است بازدید کند. فایل *HTML*، `5000 byte` بوده و به 10 آبجکت اشاره می‌کند، که دوتای آنها *(01, 02)* روی وب‌سرور ۱ هستند که فایل *HTML* نیز روی آن قرار دارد، و 8 تای دیگر *(03-010)* روی وب‌سرور ۲ هستند.
حجم فایل‌ها به این صورت است: `01=1000 byte, 02=3000 byte, 03-08 = 5000 byte, 09=010=2000 byte`
متوسط گذردهی لینک بین کامپیوتر و وب‌سرور ۱، `100000 bits/sec` و این مقدار برای لینک بین کامپیوتر و وب‌سرور ۲، `400000 bits/sec` می‌باشد. برای سادگی، فرض کنید این مقدار متناسب با سایز فایل ارسالی بین ارتباطات موازی تقسیم می‌شود.
همچنین برای وب‌سرور ۱ داریم $RTT_1 = 0.01s$ و برای وب‌سرور ۲، $RTT_2 = 0.02s$ و برای DNS داریم $RTT_{DNS} = 0.005s$.
با توجه به اینکه کاربر در ابتدا آدرس هیچکدام از وب‌سرورها را ندارد، توضیح دهید چند میلی‌ثانیه طول می‌کشد تا کاربر با استفاده از مرورگری که با هر وب‌سرور، حداکثر 5 ارتباط موازی *HTTP* از نوع `persistent` برقرار می‌کند، این صفحه وب را به صورت کامل دریافت کند.

سؤال ۴ شبکه اینترنت یک شرکت به شکل زیر را در نظر بگیرید. کاربران شبکه از سرعت پایین اینترنت شاکی هستند. درباره درخواست‌های آنان، اطلاعات زیر در دسترس است:

- *average request size* = `2 Mb`
- *average request rate* = `5 per sec`
- *Institutional router to origin server RTT* = `2 sec`

(آ) با فرض اینکه بتوان لینک متصل‌کننده به اینترنت عمومی را تعویض کرد، حساب کنید با 10 برابر شدن پنهای باند این لینک، *Access Link Utilization* چقدر تغییر می‌کند؟ *LAN Utilization* چطور؟

(ب) یک متخصص شبکه، پیشنهاد داده‌است به‌جای تغییر لینک، از یک *web cache* در شبکه استفاده شود. با توجه به اینکه *hit-rate* برای این *cache* حدودا 40% خواهد بود، حساب کنید این روش از روش قبل بهتر است یا خیر؟ اگر بهتر نیست، به ازای چه *hit-rate*ای این روش بهتر عمل خواهد کرد و اگر بهتر است، به ازای چه *hit-rate*ای روش قبل بهتر عمل خواهد کرد؟ (برای مقایسه عملکرد در این حالت، *Access Link Utilization* را مقایسه کنید.)



(ج) برای حالت اولیه و هر یک از حالات بالا، مدت تاخیر در پاسخ را حساب کنید. چه نتیجه ای می‌توان گرفت؟

سؤال ۵ در یک شبکه *File Sharing* مانند `torrent` که به صورت *P2P* کار می‌کند، توضیح دهید چگونه کاربری که تازه وارد شبکه شده می‌تواند بدون داشتن هیچ فایلی، فایل مورد نظر خود را دریافت کند؟ (راهنمایی: درباره‌ی optimistic unchoking تحقیق کنید.)

سؤال ۶ دست گرمی!

مرورگر دلخواهتان را باز کنید. نرم‌افزار `wireshark` را اجرا و در حالت *capture* قرار بدهید. یک سایت دلخواه حاوی حداقل یک تصویر را باز کنید. (ترجیحا *HTTP* باشد که به مشکل خاصی برنخورید!) پس از لود شدن کامل صفحه، به `wireshark` برگردید و آن را از حالت *capture* خارج کرده و بسته‌های *HTTP* را فیلتر کنید. سپس به سوالات زیر پاسخ دهید:

۱. مرورگر شما کدام یک از نسخه‌های *HTTP/1* یا *HTTP/1.1* را اجرا می‌کند؟ سرور چه نسخه‌ای از *HTTP* را اجرا می‌کند؟

۲. آدرس *IP* خود و سرور را مشخص کنید.

۳. آخرین زمان تغییر یکی از فایل‌هایی که از سرور دریافت کرده‌اید را مشخص کنید.

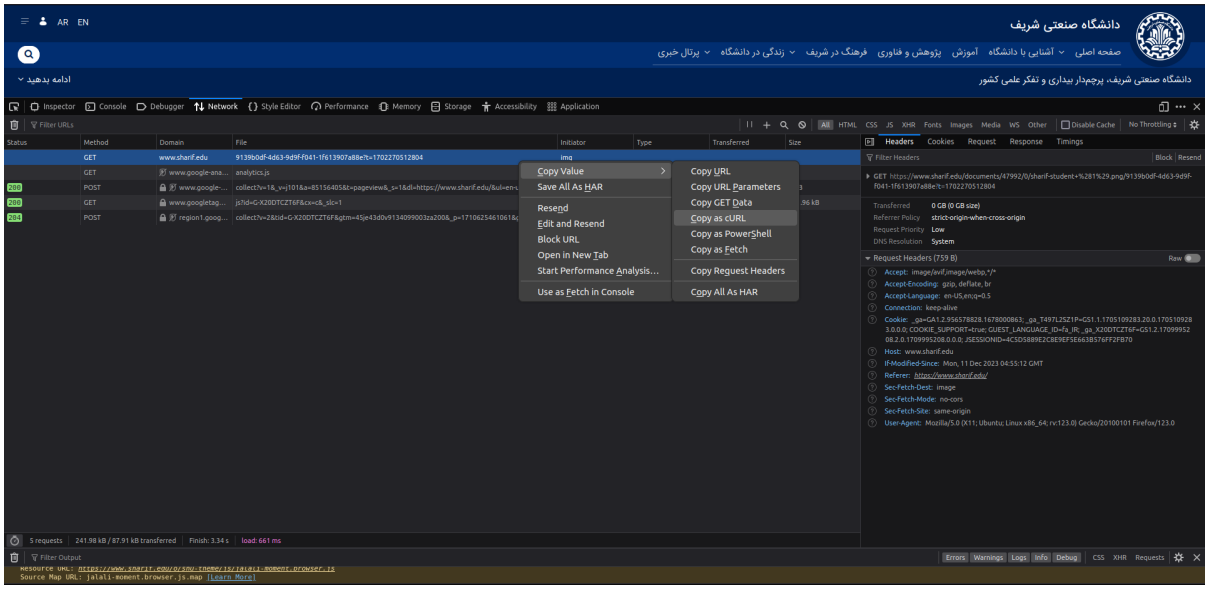
۴. بیش‌ترین حجم پیام‌های رد و بدل شده در هر لایه مربوط به کدام پروتکل است؟ (از امکانات آماری نرم‌افزار استفاده کنید و هر کدام را اعلام کنید.)

۵. محتوای پاسخ سرور را بررسی کنید. آیا سرور به صورت صریح محتوا را بازگردانده است؟

۶. اختلاف زمان بین ارسال درخواست *HTTP GET* و دریافت پاسخ *HTTP OK* چقدر است؟

۷. یکی از تصاویری که در این ارتباط از سرور دریافت شده‌اند را بازیابی کنید.

۸. با باز کردن پنجره‌ی *Web Developer Tools* در مرورگرتان، به تب *Network* رفته و معادل یکی از درخواست‌های *HTTP GET* را در قالب دستور *cURL* به‌دست بیاورید:



نقش سه مورد از *HTTP header* هایی که توسط این دستور تنظیم می‌شود را پیدا کرده و توضیح خیلی مختصری بدهید.