تمرین شماره هفت + حل و بررسی

درس شبکه های کامپیوتری – فصل سوم

مدرس: دکتر هاشمی

***

۱- می دانیم که TCP و UDP از مکمل یک برای محاسبه ی checksum بهره می برند. حال به پرسش های زیر پاسخ دهید.

الف) checksum[1] سه تا ۸ بیتی روبرو را حساب کنید. ۰۱۰۱۰۰۱۱، ۰۱۱۰۰۱۱۰ و ۱۰۰۰۱۰۱۱

01010011

+01100110

10111001

+10001011

---

1 01000100

[1] در صورت سوال مکمل یک ذکر شده بود که با این وجود مشخص بوده است که منظور محاسبه ی checksum است. از دانشجویانی که مکمل یک هر سه عدد را جداگانه نوشته اند نمره ای کسر نخواهد شد.

می دانیم که اگر حاصل جمع اعداد در مبنای دو رقم نقلی داشته باشد، باید آن را برداشته و با عدد جمع کنیم. (به این کار wrap around می گویند.) در این مثال نیز ۱ را با ۰۱۰۰۰۱۰۰ جمع می کنیم و عدد ۰۱۰۰۰۱۰۱ به دست می آید.

مکمل یک از تبدیل یک ها به صفر و صفر ها به یک به دست می آید. پس checksum در این سوال برابر با ۱۰۱۱۱۰۱۰ می باشد.

ب) دلیل استفاده از مکمل یک در این پروتکل ها چیست؟ (چرا از جمع اعداد مستقیماً استفاده نمی شود؟)

ساختار پردازنده ها به گونه ای می باشد که مکمل یک گرفتن برای آن ها ساده تر از مقایسه ی دو عدد است. پس در فرستنده مکمل یک را حساب می کنیم تا گیرنده مجبور به مقایسه ی جمع اعداد با checksum نباشد.

پ) گیرنده چگونه خطاها را شناسایی می کند؟ آیا ممکن است یک خطای یک بیتی ناشناخته بماند؟ خطای دو بیتی چطور؟

گیرنده تمامی اعداد (شامل checksum) را با هم جمع می کند. چون checksum برابر با مکمل یک جمع سایر اعداد است، پس گیرنده باید به عددی بدون رقم صفر برسد. اگر صفری وجود داشت، یعنی خطا رخ داده است.

برای درک بیش تر: در مثال بالا جمع هر سه عدد را با checksum جمع کنید!

خطای یک بیتی همواره شناسایی می شود چرا که اگر یک بیت تغییر کند، جمع اعداد نیز تغییر می کند. اما فرض کنید که رقم سمت راست یک عدد از صفر به یک و رقم سمت راست یک عدد دیگر از یک به صفر تغییر کند. آیا جمع آن دو عدد تغییر می کند؟

ت) با توجه به پاسخ خود به قسمت پ بگویید که در صورت تطابق checksum یک بسته ی UDP با محتویات آن، آیا گیرنده می تواند از بابت سالم بودن بسته اطمینان صددرصدی داشته باشد؟

چون خطای دوبیتی (و با تعداد بیت بیش تر) ممکن است ناشناخته بماند، خیر

۲- در پروتکل rdt3.0 بسته های ACKای که از گیرنده به فرستنده ارسال می شوند، شماره ترتیب (Sequence number) ندارند. چرا بسته های ACK نیازی به شماره ترتیب ندارند؟

برای پاسخ به این سوال ابتدا باید بررسی کنیم که به طور کلی چرا از شماره ترتیب استفاده می شود؟ شماره ترتیب راهکاری ست که با آن بسته های تکراری (duplicate) و بسته هایی که ترتیب شان به هم ریخته شناسایی می شوند. حال در Acknowledgment هدف این است که فرستنده بداند بسته اش به مقصد رسیده، و حال اگر چند ACK تکراری هم بیاید نه تنها مشکلی پیش نمی آید، بلکه فرستنده می فهمد که بسته ی جدید به مقصد نرسیده است. (چرا که گیرنده ACK های مربوط به بسته ی قبلی را می فرستد.)

***

۳- در طرف فرستنده ی پروتکل rdt3.0 بسته های دریافتی خراب یا با فیلد acknum نادرست، نادیده گرفته می شوند. حال فرض کنید که این پروتکل را به گونه ای تغییر دهیم که در صورت دریافت ACK ناسالم بسته ی فعلی دوباره ارسال شود. آیا پروتکل باز هم کار می کند؟

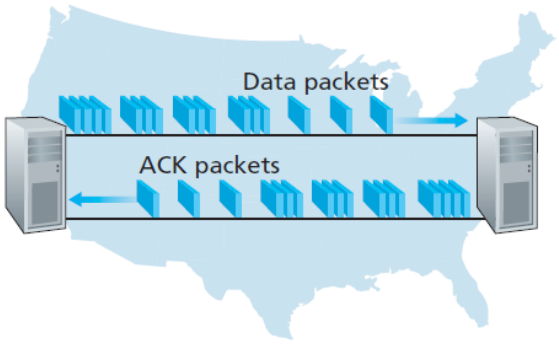
بله، کار می کند. گیرنده نمی داند که ارسال دوباره (retransmission) که فرستنده انجام داده ناشی از خراب شدن ACK بوده یا این که ACK تا زمان time out به مقصد نرسیده است. در نتیجه دوباره تاییدیه اش را می فرستد.

در حالتی که بی شمار بسته تبادل می شود این پروتکل ممکن است شبکه را ناپایدار کند، چرا که با هر خطای بیت در ACK، بسته را دوباره ارسال می کند.

۴- شکل زیر را در نظر بگیرید که در آن پروتکل انتقال در یک فاصله ی طولانی (بین دو کشور) کار می کند. اندازه ی پنجره در این پروتکل چقدر باشد تا channel utilization بیش تر از ۹۸ درصد باشد؟ فرض کنید که بسته ها ۱۵۰۰ بیتی، RTT ۳۰ میلی ثانیه و نرخ ارسال 1 Gbps باشد.

$$U_{Sender} = 0.98 = \frac{n.d_{trans}}{RTT + d_{trans}} \quad , d_{trans} = \frac{L}{R} = 1.5\,\mu sec$$

با حل معادله ی بالا به n=19600 می رسیم.



b. A pipelined protocol in operation

پرسش یکی از دانشجویان:

۱- تفاوت Push Protocol و Pull Protocol چیست؟

این یک دسته بندی برای پروتکل های client-server ای می باشد و تفاوت آن ها در معنی دو کلمه ی push و pull نهفته است. در پروتکل های pull کلاینت تغییرات را ایجاد می کند (داده را از سرور می کشد.) حال آن که در پروتکل push سرور خودش تغییراتی را در کلاینت ها ایجاد می کند و در واقع داده را به سمت آن ها *هل می دهد*. برای کسب اطلاعات بیش تر عبارت Push versus Pull Protocols را در اینترنت جستجو کنید.

۲- تفاوت میان پروتکل های POP3، IMAP و SMTP چیست؟

در متن کتاب تفاوت این ها ذکر شده، اما شاید بد نباشد مروری بر این سه پروتکل داشته باشیم. پروتکل های ذکر شده همگی پروتکل هایی در لایه ی کاربردی هستند (بسته های آن ها در درون بسته ی لایه ی انتقال جای می گیرد.) و برای جابجایی ایمیل و دسترسی به ایمیل طراحی شده اند.

POP3 یک پروتکل ساده است که سادگی آن موجب شده کارکرد آن محدود باشد. در این پروتکل نمی توان از همه جا ایمیل ها را مدیریت کرد. (مثلاً شاید بخواهیم ایمیل هایمان را در پوشه های مختلفی نگه داری کنیم) در IMAP این مسئله حل شده، و طراحی آن مناسب کاربردهای امروزی بوده که کاربران از لحاظ مکانی جابجا می شوند.

دو پروتکل IMAP و POP3 برای دسترسی به ایمیل ها طراحی شده اند، حال آن که SMTP برای جابجایی و رساندن ایمیل ها به مقصد است، و واژه ی Transfer نیز نشان دهنده ی این مطلب است.