به نام خداوند بخشنده‌ی مهربان

تمرین برنامه‌نویسی دوم[1]

شبکه‌های کامپیوتری

مدرس: مهدی خرازی بهار ۱۳۹۰

۱ مقدمه

در این تمرین شما با پروتکل TCP بیشتر آشنا شده و بخش‌هایی از آن را پیاده‌سازی خواهید کرد.

پروتکل TCP برای ارتباطات قابل اطمینان به صورت گسترده در شبکه‌ی اینترنت بکار گرفته شده است. این پروتکل با استفاده از مفهوم پورت، اتصال host-to-host که توسط IP فراهم آمده را به ارتباط end-to-end میان دو process ارتقاء داده و با مکانیزم ARQ و ارسال ACK و Checksum با احتمال بالایی وجود خطا در کانال ارتباطی را مرتفع کرده است. همچنین بکارگیری شماره‌ی ترتیب از جابه‌جایی بسته‌ها نیز جلوگیری کرده و در مجموع یک کانال مطمئن برای انتقال اطلاعات در اختیار لایه‌ی بالایی شبکه قرار داده است. بعلاوه با کمک مکانیزم‌های شناسایی و دوری از ازدحام در شبکه، TCP امکان کنترل نرخ ارسال و در نتیجه استفاده‌ی عادلانه و بهینه‌ی تمامی گره‌های متصل به شبکه از منابع موجود را فراهم می‌آورد.

هدف از این تمرین آشنایی بیشتر با مکانیزم‌های بکاررفته در پروتکل TCP بدون درگیر شدن با تمامی پیچیدگی‌ها این پروتکل می‌باشد. برای این کار شما از برنامه‌ی نوشته شده برای تمرین‌های قبلی آغاز و با اعمال تغییراتی کد مورد نیاز این تمرین را بدست می‌آورید.

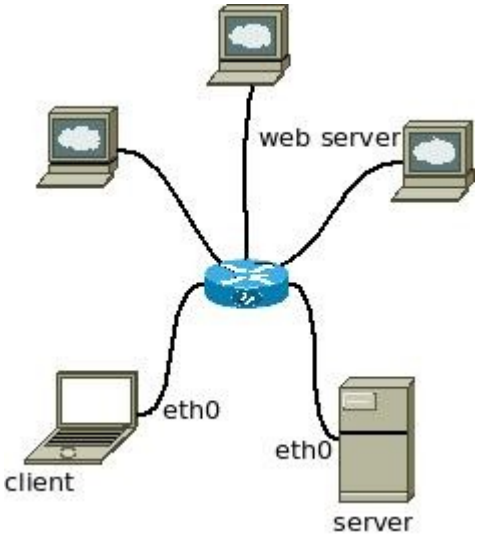
[1] با تشکر از بهنام مومنی، علی فتاح المنان، فرزانه مقدم، اشکان نیکروش، کوشا میرحسینی، امیر شیخها و حسن اسلامی

۲ محیط

اساس کار پروتکل شما در این تمرین در لایه ۴ است. به همین خاطر در این تمرین نیز، مانند تمرین قبل، از محیط پرتو[۲] برای اجرای برنامه‌ی شما استفاده خواهد شد. برای این منظور لازم است برنامه‌های خود را بر روی «چارچوب کاربر»[۳] پرتو پیاده‌سازی کنید. برای اطلاعات بیشتر در مورد این چارچوب می‌توانید به مستند «راهنمای چارچوب کاربر» مراجعه کنید.

۳ توپولوژی

توپولوژی که در اختیار هر دانشجو قرار می‌گیرد همانند شکل ۱ می‌باشد. البته برنامه‌ی شما نباید هیچگونه وابستگی به توپولوژی داشته باشد. به عنوان مثال تعداد کارگزارهای وب در توپولوژی می‌توانند در زمان آزمون تغییر کنند و یا آدرس‌های IP می‌توانند به هر شبکه‌ای تعلق داشته باشند (فرضهایی مبتنی بر range مورد استفاده در IP های شبکه همانند private بودن غیر قابل قبول هستند).



شکل ۱: توپولوژی نمونه‌ی مورد استفاده

در این توپولوژی شما دو گره‌ی client و server از شکل ۱ را پیاده‌سازی می‌کنید.

---

۲ Portable And Reliable Tool fOr Virtualization – Partov

۳ Client Framework – CF

۴ انتظارات

شما باید دو برنامه در محیط پرتو بنویسید. برنامه‌ی اول برای گره‌ی server همانند برنامه‌ای که در PA0 نوشتید و برنامه‌ی دوم برای گره‌ی client همانند برنامه‌ی PA0 با این تفاوت که به جای sokcet programming باید از محیط پرتو (مانند کد server در PA1) استفاده کنید. بر حسب نیاز می‌توانید Makefile را نیز تغییر دهید تا هر دو برنامه به طور صحیح کامپایل گردند. برنامه‌ی شما باید با اجرای فرمان make کامپایل شده و دو فایل اجرایی server.out و client.out را ایجاد نماید.

فایلهای لازم برای اجرای این دو برنامه بر روی نقشه‌ی تخصیص داده شده به همراه نسخه‌ی جدید چارچوب کاربر بر روی سایت درس قرار دارند. حتماً از نسخه‌ی نهایی چارچوب کاربر که برای PA2 آماده شده است، استفاده نمایید.

۵ برنامه‌ها

۵.۱ برنامه‌ی مربوط به client

این برنامه همانند client در تمرین‌های قبلی بوده و یک IP و یک پورت برای شروع کار در اختیار دارد. آدرس IP و پورت server در فیلد custom information مربوط به client به شکل زیر (هر یک در یک خط جداگانه) آمده است. در خط بعدی مقدار آدرس MAC مربوط به مسیریاب متصل به گره‌ی client آورده شده که client باید تمامی بسته‌های خود را از طریق این مسیریاب (Gateway) ارسال نماید. سپس مقدار window size که در اتصال TCP باید از آن استفاده نمایید (معادل با اندازه‌ی بافر مورد نیاز client) و مقدار ISN (مقدار اولیه‌ی sequence number در TCP) قرار دارند.

```
Server-IP-Address
Server-Port-Number
MAC-of-the-Gateway
Initial-TCP-Window-Size
Initial-TCP-Sequence-Number
```

همچنین برنامه‌ی client دو آرگومان برای نام host و نام فایلی که باید دانلود شود، دریافت می‌کند. به عنوان مثال برنامه‌ی client می‌تواند به شکل زیر اجرا شود:

```
./client.sh www.google.com /intl/en_com/images/srpr/logo1w.png
```

برنامه‌ی client می‌تواند قبل از فراخوانی initialize method و با پیاده‌سازی method زیر، این دو آرگومان را دریافت نماید.

```
static void SimulatedMachine::parseArguments(int argc, char *argv[]);
```

برنامه‌ی client باید بعد از اجرا این مقادیر IP و پورت server را خوانده و سپس با ارسال بسته‌های UDP به این IP و پورت و ارسال نام فایل مورد نظر (که بـه عنـوان آرگومـان گرفتـه شـده اسـت)، آدرس IP و پـورت web server نگهدارنـده‌ی فایـل درخواستی را بیابد.

یک مثال از custom information برای client مطابق زیر می‌باشد.

192.168.23.17

1234

00:17:20:00:10:A3

128

1128715

همانند PA1 محدودیتی بر روی نام بکار رفته برای ذخیره‌ی فایل دانلود شده وجود نـدارد. تنهـا لازم اسـت کـه نـام فایـل بـا کاراکترهای خاص مثل – شروع نشده و دارای پسوند صحیح (مثل html، ...) باشد.

پس از ارسال درخواست UDP و دریافت IP و پورت web server مربوطه، همانند PA0، یک اتصال TCP به آن IP و پورت ایجـاد کـرده و بـا اسـتفاده از پروتکـل HTTP فایـل مـورد نظـر را دانلـود و ذخیـره کنیـد. در ایـن تمریـن **نباید** از socket programming استفاده کنید و اتصال TCP و UDP را باید همانند برنامه‌ی server در PA1 خودتان نوشته و به طور مستقیم با header های پروتکل‌ها کار کنید.

برای اتصال TCP نیازی به پیاده‌سازی تمامی جزئیات پروتکل TCP نبوده و پیاده‌سازی موارد ذیل کفایت می‌کند:

1. Three-way TCP handshake,
2. ACK and Sequence Number,
3. Meaningful default values for headers fields and flags,
4. TCP Checksum, Reliability, and retransmission mechanisms,
5. Fast retransmission upon receiving a duplicate ACK,
6. Flow control and window size (NOT the congestion window),
7. Four-way TCP connection tear down.

۵.۲ برنامه‌ی مربوط به Server

برنامه‌ی server همانند برنامه‌ی server در PA1 بوده با این تفاوت که دیگر نیازی به پیاده‌سازی ویژگی NAT نیست. برنامه‌ی server در فیلد custom information خود اطلاعات زیر را دریافت می‌کند.

Port-for-UDP-Requests

MAC-of-the-Gateway

Count-of-Files

File-Address

IP

Port

……

برنامه‌ی server با دریافت درخواست‌های UDP بر روی پورت خط اول، نام فایل را در لیست فایلهای خود جست‌وجو کرده و سپس IP و پورت web server مربوطه را (همانند PA0) بازمی‌گرداند. یک مثال از custom information برای server در زیر آمده است:

1234

00:17:20:00:10:A3

2

/favicon.ico

213.233.171.14

8080

/old/index.html

213.233.171.15

2280

می‌توانید برای اطلاعات بیشتر راجع به پروتکل‌های مورد نیاز از لینک‌های زیر استفاده کنید.

http://en.wikipedia.org/wiki/Transmission_Control_Protocol

http://en.wikipedia.org/wiki/User_Datagram_Protocol

۶ نکات ضروری

۱. پیاده‌سازی شما به هیچ وجه نباید مبتنی بر شماره IP باشد و شما نمی‌توانید برنامه‌ی خود را به گونه‌ای بنویسید که مثلاً در آن گفته باشید «اگر فلان IP با ۲۱۳.۲۳۳ شروع شده بود فلان کار را بکن».

۲. بر حسب نیاز Makefile را تغییر دهید تا فایلهای اجرایی نهایی با نام server.out و client.out با اجرای فرمان make ایجاد شوند.

۳. شما می‌توانید IP و MAC برای هر interface را از طریق method های public کلاس Interface با نام‌های getIp و getMask بدست آورید.

۴. برنامه‌ی شما نیازی به در نظر گرفتن بسته‌های ARP ندارد. این بسته‌ها، که بین client یا server و مسیریاب‌ها رد و بدل می‌شوند، توسط محیط پرتو گرفته و بررسی آنها پیاده‌سازی شده است و برنامه‌ی شما چنین بسته‌هایی را دریافت نخواهد کرد.

۵. در صورت وجود ابهام، می‌توانید سؤالات خود را به رایانامه‌ی گروه درس به آدرس ce443@lists.ce.sharif.edu ارسال نمایید.

۷ شیوه‌ی تحویل

ابتدا برنامه‌ی خود را به طور کامل بر روی رایانه‌ی خود آزمون کرده و پس از اطمینان از صحت عمل‌کرد آن، برنامه‌ی خود را در قالب یک فایل zip بر روی سایت داوری خودکار upload کنید. این فایل zip باید شامل پوشه‌ی user و فایل Makefile باشد. پیش از ایجاد فایل zip حتما پروژه را make clean کرده تا شامل فایلهای دودویی نباشد. همچنین فایل zip نباید پوشه‌های اضافی را در بر گیرد. تنها پوشه‌ی user و فایل Makefile و نه پوشه‌ی دربرگیرنده‌ی آن‌ها را zip کرده و ارسال نمایید.