به نام خداوند بخشنده‌ی مهربان

دانشگاه صنعتی شریف
مدرس: مهدی خرازی

تمرین برنامه‌نویسی دوم*
شبکه‌های رایانه‌ای

دانشکده‌ی مهندسی کامپیوتر
بهار ۱۳۹۴

۱. هدف‌ها

- آشنایی با شبکه‌های DTN،
- آشنایی با سرآیندهای Ethernet، IP و UDP،
- آشنایی با شبکه‌های همتابه‌همتا،
- آشنایی با کدگذاری با طول متغیر.

۲. مقدمه

یکی از معماری‌های مناسب برای شبکه‌های ناپایدار و متغیر معماری DTN[1] است. این معماری در مواقعی که شبکه با اختلالات و خطای زیادی مواجه است، استفاده می‌شود. برای مثال برای ارتباطات خیلی دور که تاخیر به ساعت‌ها و یا حتی، ماه‌ها می‌رسد از DTN استفاده می‌شود. در این تمرین از شما خواسته شده است تا با استفاده از پیاده‌سازی این معماری، اطلاعات را بین شهرهایی که هرکدام یک Node به حساب می‌آیند و در فاصله‌ی دوری از هم قرار گرفته‌اند انتقال دهید.

*با سپاس از بهنام مومنی، علی فتّاح‌المنان ،مهران خلدی ،سعید محلوجی‌فر،میلاد عسگری و عرفان عبدی

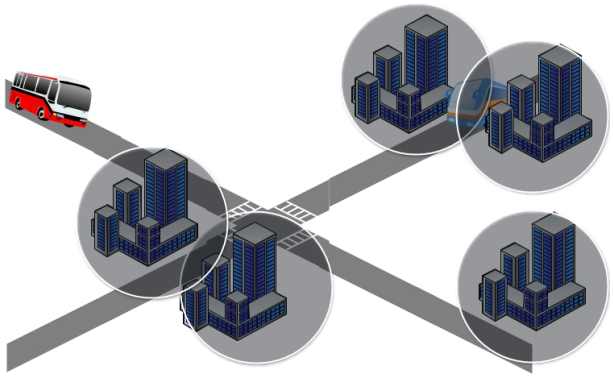
[1] Delay Tolerant Networking

۳. توضیح تمرین

همان‌طور که گفته‌شد در این تمرین قصد داریم یک DTN ساده را پیاده‌سازی کنیم. وقتی تاخیر رساندن بسته‌ها در شبکه اینترنت مدنظر نباشد می‌توان اطلاعات را به هر طریقی بین رایانه‌های مختلف در سراسر جهان منتقل کرد. همانطور که در کلاس درس فراگرفتید وقتی می‌خواهیم یک ایمیل را منتقل کنیم احتمال برخط بودن طرفین به صورت هم‌زمان بسیار کم است بنابرین اگر پس از مدت زمان محسوسی ایمیل‌ها منتقل شوند کاربران با مشکل مواجه نمی‌شوند.

در این تمرین فرض کنید شهرهای مختلفی که فاصله‌ی زیادی با هم دارند قصد دارند بسته‌هایی بین خود رد و بدل کنند. در بین این شهرها جاده هایی وجود دارد که اتوبوس‌هایی به صورت تصادفی از آن‌ها عبور می‌کنند و شهرها از برنامه‌ی زمانی این اتوبوس‌ها مطلع نیستند. شهرهای موجود می‌توانند از این اتوبوس‌ها برای ارسال اطلاعات خود استفاده کنند.

برای این منظور وقتی اتوبوسی در محدوده‌ی ارتباطی هر شهر قرار گرفت پس از برقراری ارتباط بین شهر و اتوبوس بسته‌هایی را که شهر آماده‌ی ارسال کرده، برای اتوبوس فرستاده می‌شود تا اتوبوس با گذر از شهرهای مختلف بسته را به مقصد برساند. اتوبوس‌ها نیز وقتی به محدوده‌ی شهری رسیدند تمام بسته‌هایی را که در خود دارند برای شهر مربوطه ارسال می‌کنند. دلیل این امر را می‌توان به این صورت بیان کرد: نخست آن‌که اگر بسته‌ی مخصوص همان شهر باشد وظیفه‌ی خود را انجام داده و دوم آن‌که اگر مقصد بسته، این شهر نباشد با عبور اتوبوس‌های دیگر بسته به آن اتوبوس‌ها ارسال شود و نهایتا به دست مقصد برسد.

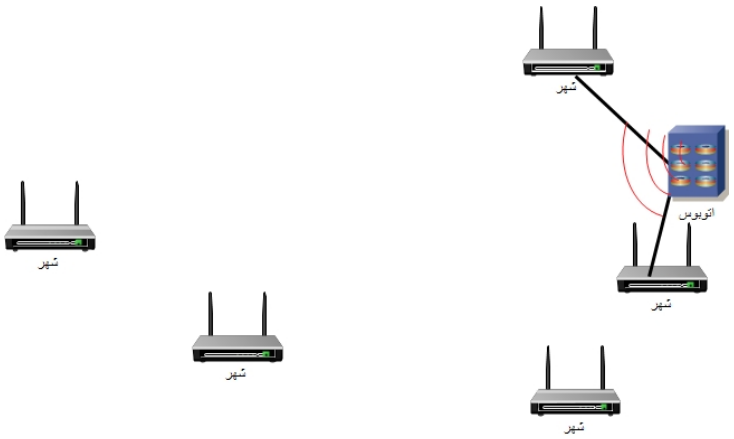


شکل ۱ : مثالی از شهرها و جاده‌های ارتباطی آن‌ها

۴. پیاده سازی

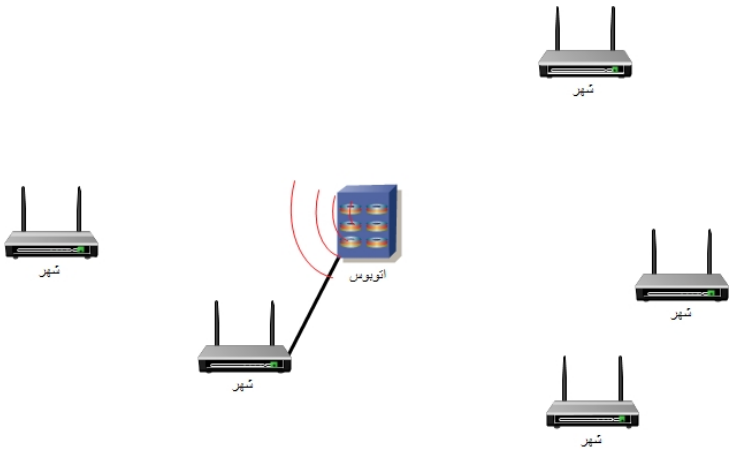
برنامه‌ای که شما می‌نویسید برنامه‌ی مربوط به اتوبوس‌ها و شهرها می‌باشد. در این قسمت باید از شهرهایی که به اتوبوس متصل هستند، آگاه شوید تا به تبادل بسته ها بپردازید. از طرفی، با وارد کردن دستور `walk` در اتوبوس، حرکت صورت می‌گیرد که می‌تواند وضعیت ارتباطی اتوبوس و شهرها را تغییر دهد.

لازم به ذکر است که هر کدام از اتوبوس‌ها و شهرها یک Node به حساب می‌آیند. همانند شکل ۲ وقتی اتوبوسی در محدوده‌ی شهر قرار می‌گیرد می‌تواند با آن ارتباط برقرار کند. در این شکل همان‌طور که دیده می‌شوند اتوبوس از طریق تنها واسط[۲] خود با دو شهر موجود در محدوده‌ی خود، ارتباط برقرار کرده است.



شکل ۲: اتصالات شبکه در زمان t_1

در شکل ۳ نیز وضعیت اتصال اتوبوس به شهرها در زمان t_2 نشان داده شده است. این وضعیت پس از حرکت اتوبوس در اتفاق افتاده است. همان‌طور که دیده می‌شود در هر واحد زمانی وضعیت اتصال اتوبوس به شهرها تغییر می‌کند.



شکل ۳: اتصالات شبکه در زمان t_2

[۲] Interface

در ادامه به بررسی پیاده‌سازی برنامه‌ی اتوبوس و شهر می‌پردازیم.

۱.۴ اتوبوس

در این قسمت طریقه پیاده‌سازی برنامه‌ی اتوبوس و جزئیات آن بررسی‌شده است.

۱.۱.۴ Broadcast

در ابتدای کار، اتوبوس‌ها اطلاعی از آدرس شبکه‌های شهرهای مختلف ندارند. بنابراین باید در هر مرحله پس از حرکت اتوبوس لیست Network IP شهرهای موجود در محدوده‌ی اتوبوس را به دست آوریم. برای این کار بسته‌ای UDP با محتوای[۳] خالی Broadcast می‌کنیم و از روی پاسخ‌هایی که دریافت می‌شود، لیست آدرس Network IP شهرهای متصل را به دست می‌آوریم. بسته‌ای که به عنوان Broadcast ساخته می‌شود از پورت `5000` اتوبوس برای پورت `5000` شهر ارسال می‌شود.

این عمل با وارد کردن دستور `scan` انجام می‌شود. پس از دریافت جواب این بسته، اتوبوس باید قادر باشد آدرس IP شبکه شهرها را به صورت زیر چاپ کند.

```
City with [Network IP] is in range
```

که در آن Network IP آدرس IP شبکه شهر می‌باشد. به عنوان مثال، عبارت زیر یک عبارت معتبر است:

```
City with 192.168.123.0/24 is in range
```

۲.۱.۴ ارسال اطلاعات اتوبوس

پس از آگاه شدن اتوبوس از شهرهای قابل رویت، نوبت به تبادل اطلاعات می‌رسد. برای این منظور نیاز داریم تا بسته‌ای حاوی اطلاعات موجود در اتوبوس بسازیم و برای شهرها ارسال نماییم. پس از وارد کردن دستور `sync` تمامی پیام‌های اتوبوس برای شهرهای مجاور اتوبوس ارسال می‌شوند.

فرض می‌کنیم پیام‌هایی که هر شهر برای شهر دیگر در نظر می‌گیرد مانند جدول ۱ است.

در بسته‌هایی که بین اتوبوس و شهرها مبادله می‌شود نیز بعد از سرآیند UDP، تعداد پیام‌ها که عددی ۴ بایتی است و بعد از آن، پیام‌ها مانند جدول ۲ می‌آیند.

جدول ۱: ساختار داخلی پیام‌های بین شهرها

2 Bytes	4 Bytes	4 Bytes	1 Byte	1 Byte	Variable Bytes of data
Length	Source IP	Destination IP	TTL	ID	Message

[۳] Payload

در جدول ۱ Length طول کلی پیام که یک عدد ۲ بایتی، Source IP آدرس IP مبدا پیام و Destination IP آدرس IP مقصد پیام می‌باشد. این مقادیر اعدادی ۴ بایتی هستند. TTL مدت زمان معتبر بودن پیام است که یک عدد ۱ بایتی است. جزییات استفاده از آن در ادامه خواهد آمد. ID شناسه پیام است که برای متمایزسازی پیام‌ها از آن استفاده می‌شود. این مقدار عددی ۱ بایتی خواهد بود. در نهایت نیز پیامی که قرار است منتقل شود خواهد آمد. این مقدار یک رشته خواهد بود.

جدول ۲ : ساختار بسته‌های تبادل پیام بین شهر و اتوبوس

Header
Number of messages
Message 1
...
Message 2

*نکته:*توجه کنید که وجه تمایز پیام‌ها در این تمرین، مقادیر ID و IP مبدا هر پیام خواهد بود.

اتوبوس بسته‌های خود را با پورت `2000` برای پورت `8000` شهرها ارسال می‌کند و شهرها نیز بسته‌های خود را با پورت `8001` برای پورت `3000` اتوبوس می‌فرستند. پس از ارسال هر بسته که حاوی چند پیام می‌باشد، این اطلاعات به صورت زیر چاپ می‌شود:

```
[Numbers] messages are sent to [City IP]
```

که در آن Numbers تعداد کل پیام‌هایی است که برای هر شهر ارسال شده‌است.

توجه داشته باشید که اگر بسته‌ای حاوی پیامی باشد که اتوبوس آن را به مقصد رسانده است، باید از جدول پیام‌هایی که اتوبوس حمل می‌کند، حذف شود. برای این منظور شهرها در جواب بسته Broadcast علاوه بر Network IP، Subnet Mask خود را نیز ارسال می‌کنند(جزییات این بسته در ادامه خواهد آمد). اتوبوس با استفاده از این اطلاعات و محتوای بسته که شامل IP مقصد هر پیام می‌باشد، اطلاعات را از پایگاه داده خود حذف می‌کند(فرض می‌شود هر بسته‌ای که اتوبوس ارسال می‌کند، لزوما به شهر می‌رسد). توجه کنید این عمل بعد از دریافت بسته پیام‌های شهر(که می‌تواند بسته‌ای حاوی صفر پیام باشد) و قبل از کم کرد TTL انجام می‌شود.

در این صورت مقدار زیر در خروجی چاپ می‌شود:

```
Message with dst [IP] and src [IP] arrived to its destination
```

۳.۱.۴. محتویات اتوبوس

در هر لحظه برنامه شما باید قادر باشد اطلاعات بسته‌هایی را که در اتوبوس موجود دارید نمایش دهد. در این حالت باید پس از وارد شدن دستور `print` اطلاعات بسته‌های خود موجود را در خروجی به صورت زیر چاپ کنید:

```
Dst IP: [IP] | Src IP: [IP] | ID: [ID] | TTL: [TTL] | [Message]
```

که در آن ID مشخصه پیام و Message متن پیام می‌باشد. اگر پایگاه داده خالی بود عبارت زیر در خروجی چاپ می‌شود:

```
No message
```

۲.۴. شهر

در این قسمت جزییات پیاده‌سازی شهر بررسی می‌شود.

۱.۲.۴. پاسخ بسته Broadcast

همان‌طور که پیش‌تر ذکر شد در این بسته که از پورت `5000` شهر برای پورت `5000` اتوبوس به صورت Unicasted ارسال می‌شود، Network IP و Subnet Mask شهر مورد نظر به عنوان محتوای بسته UDP، قرار می‌گیرند. جدول ۳ ساختار این بسته‌ها را نشان می‌دهد.

جدول ۳: ساختار پاسخ بسته‌ی Broadcast

Header
Subnet Mask
Network IP

*نکته:*توجه داشته باشید که در ادامه برای ارسال بسته‌ها نیاز به Mac Address این شهر خواهید داشت؛ پس نیاز است پس از دریافت این بسته توسط اتوبوس، این اطلاعات را ذخیره کنید.

۲.۲.۴. ارسال اطلاعات شهر

پس از دریافت بسته‌ی حاوی پیام‌های اتوبوس، با بررسی شرط تمایز پیام‌ها که در قسمت اتوبوس ذکر شد، پیام‌هایی را که در پایگاه داده شهر نیست به آن اضافه می‌کنیم. پس از اضافه کردن پیام‌های ناموجود، پیام‌هایی را که در شهر موجود است ولی در پایگاه داده اتوبوس یافت نمی‌شود را به همراه شکلی که در جدول ۱ ذکر شد، برای اتوبوس ارسال می‌کنیم. پس از ارسال بسته‌ی حاوی پیام‌ها مقدار زیر در خروجی چاپ می‌شود:

```
[Numbers] messages are sent to [Bus IP]
```

که در آن Numbers تعداد پیام‌های ارسالی به اتوبوس خواهد بود.

۳.۲.۴. دریافت اطلاعات از اتوبوس

شهر پس از دریافت پیام‌ها از شهر مقدار زیر را در خروجی چاپ می‌کند:

```
[Numbers] messages are received from [Bus IP]
```

توجه کنید اگر مقصد پیامی خود شهر بود، باید آن پیام را به پایگاه داده مرکزی خود اضافه نکند و مقدار زیر را در خروجی چاپ کند:

```
A message received which is owned by this city
```

۴.۲.۴. ساخت پیام

هر شهر می‌تواند پیام‌هایی را در پایگاه داده خود ایجاد کند تا پس از رسیدن اتوبوس آن‌ها را ارسال نماید. برای این منظور، پس از وارد کردن دستور `msg` مقادیر موجود در پیام را به ترتیب زیر وارد می‌کنیم. در ابتدا آدرس IP مبدا پیام وارد می‌شود. پس از آن آدرس IP مقصد را وارد می‌کنیم. سپس ID آن پیام و در نهایت مقدار Message آن پیام که ممکن است حاوی فاصله نیز باشد وارد می‌شود. توجه کنید که لزوما IP مبدا پیام ایجاد شده، IP همان شهری که این پیام را ایجاد می‌کند، نخواهد بود.

*نکته:*تمامی این دستورات در یک خط وارد می‌شود.

۵.۲.۴. محتویات شهر

در هر لحظه برنامه شما باید قادر باشد اطلاعاتی را که در پایگاه داده‌ی شهر موجود است را نمایش دهد. در این حالت باید پس از وارد شدن دستور `print` اطلاعات پیام‌های موجود را مانند زیر در خروجی چاپ کنید.

```
Dst IP: [IP] | Src IP: [IP] | ID: [ID] | TTL: [TTL] | [Message]
```

که در آن ID مشخصه پیام و Message متن پیام می‌باشد. اگر پایگاه داده خالی بود عبارت زیر در خروجی چاپ می‌شود:

```
No message
```

نکته مهمی که بایستی بدان توجه کرد TTL هر پیام است. این ویژگی برای جلوگیری از میل کردن تعداد پیام‌های شهرها و اتوبوس‌ها به سمت بی‌نهایت پیش‌بینی شده است. این مقدار از طریق Custom Information موجود در شهرها به شما داده می‌شودو با استفاده از تابع `getCustomInformation()` قابل استفاده است. در رابطه با TTL باید به نکات زیر توجه کرد:

۱. اتوبوس قبل از فرستادن پیام‌ها به شهرها از TTL پیام نمی‌کاهد.

۲. در نتیجه شهر اطمینان خواهد داشت که TTL پیام ≥ 1 خواهد بود.

۳. شهر بعد از فرستادن پیام‌های خود به اتوبوس از TTL تمامی پیام‌های موجود در پایگاه داده خود یک واحد می‌کاهد.

۴. اتوبوس پس از دریافت پیام‌ها از شهر و بعد از اضافه کردن به پایگاه داده‌ی خود، از TTL تمامی پیام‌ها می‌کاهد.

۵. اگر اتوبوس به k شهر به‌طور هم‌زمان متصل باشد، TTL پیام‌های موجود در هر شهر تنها یک‌بار کاهش می‌یابد. دلیل این امر آن است که هر شهر تنها یک‌بار به اتوبوس جواب می‌دهد(مورد ۳). ولی، TTL پیام‌های موجود در پایگاه داده اتوبوس k بار کاهش می‌یابد که دلیل آن دریافت k جواب از شهرهای مختلف است (مورد ۴).

۶. این باعث می‌شود که TTL پیام‌های اتوبوس به ترتیب دربافت پیام‌ها از شهرهای متصل به اتوبوس وابسته باشد.

*نکته:*در این تبادل، TTL هر پیامی که مبادله می‌شود دقیقا یک‌بار کاهش می‌یابد. در رابطه با TTL توجه به مثال زیر سودمند است:

۱. فرض کنید اتوبوس به دو شهر city0 و city1 متصل است.

۲. فرض کنید اتوبوس ۳ پیام (m1)، (m2) و (m3) را در پایگاه داده خود دارد.

۳. از طرفی city0 پیام‌های (m4) و (m5) و city1 پیام (m6) را در پایگاه داده خود دارند.

۴. TTL را برای تمام این پیام‌ها ۵ در نظر بگیرید.

۵. اتوبوس پیام‌های (m1,m2,m3) را به city0 و city1 ارسال می‌کند.

۶. city0 این پیام‌ها را دریافت و (m4,m5) را به اتوبوس ارسال می‌کند و TTL را برای (m1,m2,m3,m4,m5) در پایگاه داده خود برابر ۴ قرار می‌دهد.

۷. city1 نیز ۳ پیام دریافت می‌کند و با (m6) به آن جواب می‌دهد. در نهایت نیز TTL را برای (m1,m2,m3,m6) در پایگاه داده خود برابر ۴ قرار می‌دهد.

۸. اتوبوس بسته‌ی پیام‌ها را از city0 دریافت می‌کندو TTL را کاهش می‌دهد. حال اتوبوس (m1,m2,m3,m4,m5) را با TTL ۴ در پایگاه داده خود دارد.

۹. اتوبوس بسته‌ی پیام‌ها را از city1 دریافت می‌کند و TTL را کاهش می‌دهد. حال، داده‌های موجود در پایگاه داده اتوبوس (m1,m2,m3,m4,m5,m6) است و TTL برای (m1,m2,m3,m4,m5) برابر ۳ و برای (m6) برابر با ۴ است.

۵. اطلاعات تکمیلی

۱.۵ Custom Information

تنها مقداری که در این قسمت قرار می‌گیرد و شما می‌توانید از آن استفاده کنید، مقدار TTL پیام‌ها خواهد بود.

۲.۵ ترتیب پیام‌ها در پایگاه داده

همان‌طور که گفته شد، Source IP و ID مقادیری هستند که باعث تمایز داده‌ها می‌شوند. شما باید پیام‌ها را بر اساس Source IP و در صورت برابر بودن آن‌ها بر اساس ID به صورت صعودی مرتب کنید و در پایگاه‌های داده مرکزی اتوبوس‌ها و شهرها نگه‌داری کنید. در نتیجه، نخست آن‌که در صورت وارد شدن دستور `print` چه در شهر و چه در اتوبوس، ترتیب چاپ پیام‌ها براساس اصل گفته شده خواهد بود. دوم آن‌که در صورت وارد شدن دستور `sync` در اتوبوس ترتیب تبادل پیام‌ها از اتوبوس به شهر و از شهر به اتوبوس برهمین اساس خواهد بود.

۶. نکات ضروری

- به علت اینکه نمره‌ی تمرین به صورت خودکار داده می‌شود، ساختار پیام‌های گفته شده باید دقیقاً به صورت گفته شده باشد.
- نحوه‌ی ارزیابی ممکن است دچار تغییراتی شود و تست‌های دیگری اضافه شوند.
- در صورتی‌که هر مشکل یا پرسشی داشتید که فکر می‌کنید پاسخ آن برای همه مفید خواهد بود، آن را به گروه اینترنتی درس ارسال کنید.
- از فرستادن جواب تمرین به گروه اینترنتی درس خودداری کنید.
- تمام برنامه‌ی شما باید توسط خود شما نوشته شده باشد. فرستادن کل یا قسمتی از برنامه‌تان برای افراد دیگر، یا استفاده از کل یا قسمتی از برنامه‌ی فرد دیگری، حتی با ذکر منبع، تقلب محسوب می‌شود.
- پس از اتمام کارتان لازم است پوشه‌های user-bus و user-city را به همراه Makefile فشرده کرده (می‌توانید این کار را با اجرای دستور `make submit` انجام دهید) و از طریق وب‌سایت پرتو ارسال نمایید.