به نام خداوند بخشنده مهربان

دانشگاه صنعتی شریف
مدرس: مهدی خرازی

تمرین برنامه نویسی دوم *
شبکه‌های رایانه‌ای

دانشکده‌ی مهندسی کامپیوتر
پاییز ۱۳۹۵

اهداف تمرین

- آشنایی با شبکه‌های P2P و شبکه‌ی بیت‌تورنت[۱]
- آشنایی با پروتکل بیت‌تورنت
- آشنایی با پروتکل HTTP
- آشنایی با روش CRC برای آزمون سلامت داده
- کار با سرآیند TCP
- آشنایی با روش RFC 793

۱. مقدمه

در درس با شبکه‌های همتابه‌همتا[۲] و یکی از مهم ترین و کاربردی ترینِ این شبکه‌ها در حال حاضر، بیت‌تورنت آشنا شدید. شبکه‌ی همتا به همتای بیت‌تورنت برای دریافت و یا اشتراک‌گذاری پرونده و داده‌های معمولاً حجیم است. در این شبکه هر همتا همزمان در حال ارسال و دریافت بخشی از داده است، به این ترتیب بار ارسالی و دریافتی بین همتاها تقسیم می‌شود. برای هماهنگی اولیه و پیداکردن یکدیگر بین همتاها از ردیاب‌ها استفاده می‌شود. ردیاب[۳] معمولاً کارگزار مستقلی است که فهرست همتای هر تورنت با اطلاعات دسترسی به آنها را دارد.

مهم ترین اهداف شبکه‌ی بیت‌تورنت عبارتند از:

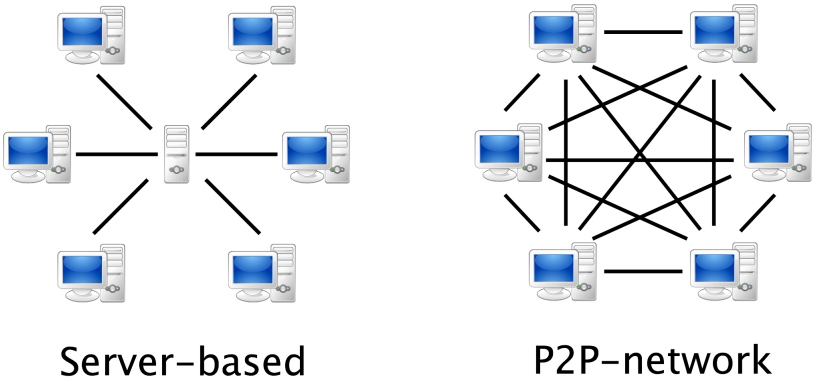
* با سپاس از سولماز سلیمی، رضا میرعسگر شاهی، پارسوآ خورسند، مهدی بهروزی‌خواه، پیمان عزتی

[۱] BitTorrent
[۲] Peer to Peer
[۳] Tracker

- تقسیم بار ترافیکی(برداشتن ترافیک سنگین از یک کارگزار یا تعداد کم کارگزار)
- تقسیم فضای ذخیره‌سازی بین همتاها
- افزایش ضریب اطمینان وجود داده و دسترسی به آن
- افزایش ضریب اطمینان از صحت داده و بررسی سریع‌تر و آسان‌تر صحت

۲. بیت‌تورنت

پروتکل بیت‌تورنت توسط Bram Cohen در سال ۲۰۰۱ معرفی شد. در این پروتکل ابتدا همتایی که قصد اشتراک گذاری (تعدادی) پرونده را دارد، پرونده تورنت[۴] می‌سازد (این پرونده معمولاً با پسوند torrent. ذخیره می‌شود)، که مشخصات مورد نیاز برای دیگر همتاها را فراهم می‌کند. همتای اولیه[۵] (همتایی که قصد اشتراک داده ها را دارد) پرونده‌ها را به تعدادی قطعه[۶] تقسیم می‌کند. برای هر قطعه با SHA-1[۷] یک درهم‌ساز[۸] به طول ۲۰ بایت می‌سازد. از جمله اطلاعاتی که در پرونده تورنت ذخیره می‌شود، نام پرونده‌ها، تعداد قطعه‌ها و مقدار درهم‌ساز هر قطعه است. در پرونده تورنت آدرس ردیاب‌ها نیز قرار می‌گیرد؛ در ادامه درمورد ردیاب‌ها توضیح داده خواهد شد. سپس این پرونده تورنت منتشر می‌شود و همتاهای دیگر به وسیله‌ی اطلاعات آن می‌توانند به یکدیگر متصل شوند.



شکل ۱: شِمای کلی شبکه‌های مبتنی بر کارگزار و شبکه‌های همتا به همتا

۱.۲ ردیاب

آدرس ردیاب در فایل تورنت مشخص می‌شود. هر همتا با اتصال به ردیاب و ارسال مشخصات خود و پرونده تورنت، از ردیاب در قالب یک HTTP درخواست گرفتن آدرس و مشخصات همتاهای دیگر در شبکه‌ی این پرونده تورنت

[۴] Torrent file
[۵] Initial Peer
[۶] Piece
[۷] یک روش رمزنگاری پر استفاده
[۸] Hash code

را می‌کند. ردیاب هم در قالب یک HTTP پاسخ می دهد.

سپس همتا با داشتن اطلاعات همتاهای دیگر به آنها درخواست اتصال می دهد و بعد از اتصال شروع به دریافت و ارسال داده می‌کنند.

مقادیر URL درخواست 'GET' به ردیاب عبارتند از:

`Info_hash:`

یک کد درهم‌ساز ۲۰ بایتی از اطلاعات تورنت، این کد شناسه‌ی هر تورنت است.

`Peer_ID:`

یک شناسه یکتا برای هر همتا. این شناسه توسط همتا تولید می‌شود.

`Port:`

پورتی که همتا روی آن گوش می‌دهد. معمولاً پورت‌های ۶۸۸۱ تا ۶۸۸۹ به عنوان پورت‌های پروتکل بیت‌تورنت شناخته و استفاده می‌شوند.

`uploaded:`

مجموع مقداری از اطلاعات به بایت که همتا به دیگر همتاها ارسال کرده است. این مقدار به صورت دهدهی و ASCII ثبت می‌شود.

`downloaded:`

مجموع مقداری از اطلاعات به بایت که همتا از دیگر همتاها دریافت کرده است. این مقدار به صورت دهدهی و ASCII ثبت می‌شود.

`left:`

مجموع مقداری از اطلاعات به بایت که همتا نیاز دارد تا داده‌اش کامل شود. این مقدار به صورت دهدهی و ASCII ثبت می‌شود.

`compact:`

مقدار صفر یا یک میگیرد. مقدار یک یعنی که همتا قابلیت پشتیبانی (تحلیل) از پاسخ ردیاب به صورت فشرده را دارد. در پاسخ فشرده به جای فهرست همتاها به صورت متن، یک رشته از بایت‌ها به عنوان فهرست همتاها داده می‌شود. ۶ بایت به ازای هر همتا، ۴ بایت اول آدرس و ۲ بایت بعد پورت، هر کدام به صورت Network endian قرار می‌گیرند.

`event:`

این فیلد باید یکی از ۳ مقدار started، stopped و completed را بگیرد. اگر خالی گذاشته شود نادیده گرفته می‌شود. started در اولین بسته یا بعد از بسته‌ی HTTP با واقعه‌ی stopped یا بعد از مدت طولانی نفرستادنِ بسته به ردیاب فرستاده می‌شود. با فرستادن این واقعه شما در فهرست همتاهای فعال آن ردیاب برای آن تورنت قرار می‌گیرید. stopped برای خروج از لیست همتا های فعال استفاده می‌شود و completed برای اعلام داشتن تمام داده است.

`IP:`

این فیلد معمولا نیاز نیست، چون می‌توان آی‌پی همتا را از لایه‌ی شبکه‌ی بسته‌ی دریافتی درآورد. ولی زمانی که همتا از طریق پیشکار [۹] یا هر روشی که ممکن است باعث مشخص نشدن آی‌پی واقعی همتا شود برای اتصال به ردیاب استفاده می‌کند، این مقدار داده می‌شود.

`Numwant:`

تعداد همتاهایی که همتا ترجیح می‌دهد در پاسخ دریافت کند. مقدار پیش‌فرض معمولاً ۵۰ است و در این صورت نیازی به این فیلد نیست.

`Key:`

یک متغیر اختیاری دیگر که حاوی شناسه ایست که با دیگر همتاها به اشتراک گذاشته نمی‌شود. معمولاً برای شناسایی همتاست. برای مثال زمانی که آی‌پی همتا عوض می‌شود.

پاسخ ردیاب به صورت HTTP response است و داری مقادیر زیر است:

`Failure_reason:`

اگر این فیلد قرار داده شود، دیگر بقیه فیلدها نمی‌آیند. مقدار این فیلد معمولاً یک رشته‌ی ASCII است که توضیح می‌دهد دلیل خطا چیست.

`Warning_message:`

مانند خطای Failure است امّا همچنان بقیه فیلدها به صورت عادی تنظیم می‌شوند.

`Interval:`

فاصله‌ی زمانی که به همتا گفته می‌شود بین درخواست‌هایش لحاظ کند. یعنی همتا این مقدار صبر می‌کند تا درخواست بعدی را بفرستد.

سپس فهرست همتاها و مشخصات اتصال به آنها می‌آید.

۲.۲ همتا

همتاها با اتصال به ردیاب و ارسال مشخصات لازم، دیگر همتاهای آن تورنت را از ردیاب دریافت می‌کنند. حال می‌توانند قطعه‌هایی که را که ندارند از دیگر از همتا درخواست نمایند و یا قطعات درخواستی دیگر همتاها را به آنها ارسال نمایند. هر اتصال بین همتاها متقارن و دوطرفه است. وقتی یک همتا یک قطعه را کامل دریافت کرد، کد درهم‌ساز آن را بررسی می‌کند و اگر مطابق بود، باید داشتن آن قطعه را به همه همتاها اعلام کند. در هر ارتباط هر سر ارتباط یا choked هستند یا not choked، یا interested هستند یا not interested. هر اتصال با وضعیت choked و not intersted شروع می‌شود. زمانی داده انتقال پیدا می‌کند که ارسال کننده به حالت choked not و دریافت کننده به حالت interested باشد. پیغام handshake اولین پیغام بین همتاها است، این پیغام شامل نام پروتکل، تعدادی بایت رِزِرو، info hash و peer ID است. در پاسخ handshake، همتای مقابل یک handshake مشابه ارسال می‌کند.

[۹] Proxy

پیغام‌ها:

```
HandShake: <pstrlen><pstr><reserved><info_hash><peer_id>
```

مشخصات هر بخش پیغام HandShake بدین شکل است:

pstrlen: یک بایت حاوی طول pstr به بایت.

pstr: یک رشته[10] حاوی نام کامل پروتکل.

reserved: حاوی ۸ بایت رزِرو شده.

info_hash: ۲۰ بایت حاوی کد در هم‌ساز ساخته شده با SHA-1 از تورنت؛ این کد برای شناسایی تورنت‌ها استفاده می‌شود.

peer_id: ۲۰ بایت حاوی شناسه‌ی همتای ارسال کننده پیغام.

تمام پیغام‌های دیگر پروتکل بیت‌تورنت به صورت <length prefix><message ID><payload> هستند. که طول lenght prefix چهار بایت و message ID یک بایت است. یعنی همیشه طول پیغام‌ها حداقل ۵ بایت است(به جز پیغام کاملا خالی که صرفا برای اعلام زنده بودن اتصال است، این پیغام ۴ بایت صفر برای نشان دادن طول صفر پیغام است). بسیاری از پیغام‌ها بدون پایه‌بار[11] هستند یعنی دقیقاً ۵ بایت طول دارند.

```
Keep alive: <len=00 00 00 00>
```

این پیغام برای جلوگیری از بسته شدن اتصال توسط همتای مقابل می‌باشد. اگر همتا تا مدتی هیچ پیغامی دریافت نکند ممکن است فرض کند که در اتصال مشکلی پیش آمده و اتصال خود را قطع کند.

```
Choke: <len=00 00 00 01><id=00>
```

اعلام choke

```
Unchoke: <len=00 00 00 01><id=01>
```

اعلام unchoke

```
Interested: <len=00 00 00 01><id=02>
```

اعلام interested

```
Not interested: <len=00 00 00 01><id=03>
```

اعلام interested not

[10] String

[11] Payload

```
Have: <len=00 00 00 05><id=04><piece index>
```

اعلام داشتن یک قطعه (معمولا بعد از دریافت آن)

```
Bitfield: <len=00 00 00 01 + X><id=05><bitfield>
```

این پیغام برای این است که اعلام کند کدام قطعات را همتا دارد، و معمولاً بعد از handshake ارسال می‌شود. به این صورت که در bitfield به تعداد قطعات بیت می‌آید. و یعنی X خارج قسمت تعداد قطعات بر ۸ (به اضافه‌ی ۱ اگر باقیمانده داشته باشد) است. هر بیت اگر یک باشد یعنی آن بیت را دارم و هر بیت صفر به معنی عدم داشتن آن قطعه است.

```
Request: <len=00 00 00 0d><id=06><index><begin><length>
```

برای در خواست قطعه از همتای دیگر. index شماره قطعه، begin شماره بایتی از قطعه که از آنجا درخواست دریافت داریم و length تعداد بایتی که بعد از آن بایت می‌خواهیم. برای مثال فرض کنید کل قطعه اول، که طولش ۱۰۲۴ بایت است را می خواهیم. مقادیر را این‌گونه تنظیم می‌کنیم:

index: 0

begin: 0

length: 1024

```
Piece: <len=00 00 00 09 + X><id=07><index><begin><block>
```

ارسال داده‌ی قطعه به همتا. که index شماره قطعه، begin بایت شروع در قطعه و ،block داده است.

```
Cancel: <len=00 00 00 0d><id=08><index><begin><length>
```

لغو درخواست داده. که index شماره قطعه، begin بایت شروع در قطعه و length طول داده است.

```
Port: <len=00 00 00 03><id=09><listen-port>
```

اعلام پورتی که روی آن گوش می‌دهد.

۳. صورت تمرین

در این تمرین شما باید وظایف همتا شبکه‌ی بیت تورنت و ردیاب تورنت را شبیه سازی کنید. خواندن از پرونده tor-rent و الگوریتم‌های بارگیری و بارگذاری[۱۲] **جزء تمرین نیستند.** اطلاعات پرونده تورنت در custom information در اختیار شما قرار می‌گیرد. شما باید به ردیاب درخواست را ارسال کنید. درخواست را گرفته، آن را تحلیل کنید.

[۱۲] Download and Upload

سپس به همتاهای دیگر متصل شوید و داده‌ها را ارسال و دریافت کنید و اطلاعت خود را به کمک ردیاب به‌روز کنید. در بخش ردیاب شما باید اطلاعات هر همتا را بعد از درخواست نگه دارید. اطلاعات همه‌ی همتاهایی که در بانک اطلاعاتی خود دارید را برای این همتا ارسال کنید.

- تک تک قسمت‌ها را باید مطابق با استاندارد IPv4 پر کنید. [۱۳]
- دقت کنید که هر گره یک `Interface` دارد، آدرس مک مبدا و آدرس آی‌پی مبدا را از روی اطلاعات آن تنظیم کنید. آدرس مک مقصد، آدرس مک گره‌ی بعدی باید باشد، که این آدرس در در انتهای `Custom Information` آمده است.
- `Protocol` پروتکل لایه بالایی را باید در این قسمت قرار دهید. چون تمام بسته‌ها TCP هستند، پس با `6` پر کنید.
- `IHL` ، `Version` ، `Total Length` را با استاندارد آی‌پی نسخه ۴ پر کنید.
- بقیه قسمت‌ها را با صفر پر کنید.

۱.۳. سرآیند TCP

در تمام بسته‌های این تمرین از سرآیند TCP استفاده می‌شود، از آنجایی که بحث برقراری اتصال، مدیریت بسته‌های گم شده، مدیریت ترافیک و ... در این تمرین نیست نیاز به پر کردن برخی فیلدهای این سرآیند نیست.

0 – 7	8 – 15	16 – 23	24 – 31
Source port		Destination port	
Sequence number			
Acknowledgment number			
Data offset	Reserved	Flags	Window size
Checksum		Urgent pointer	
Payload . . .			

شکل ۲: سرآیند کامل TCP

[۱۳] https://en.wikipedia.org/wiki/IPv4#Header

پورت مبدا و مقصد باید به درستی بر اساس نیاز پرشوند، sequence number و Acknowledgment number را همیشه **صفر** بگذارید. مقدار Data Offset را ۵ بگذارید (طول سرآیند TCP به واحد ۴ بایت)؛ توجه کنید که طول این فیلد ۷ بیت است و یک بایت کامل نیست. فیلدهای reserved و flags و window size را هم صفر بگذارید. checksum

را صفر بگذارید. urgentpointer هم صفر بگذارید.

۲.۳. وظایف ردیاب

در ابتدای costum information اگر عبارت `Tracker Info:` نوشته شده بود، یعنی شما یک ردیاب هستید و باید وظایف ردیاب را پیاده سازی کنید. در این راستا در ادامه information lrcustom تعداد فایل‌های تورنت آپلود شده و بعد hash lrinfo هر کدام را مشاهده می‌کنید. متن ۱، نمونه‌ی از محتوای custom information برای یک ردیاب است.

```
Tracker Info:
Listening port: 31259
3
Info hash 1: 4de0
Info hash 2: 244f
Info hash 3: 01ca
The Mac Address: 14:32:F3:A2:10:00
```

متن ۱، نمونه‌ای از داده‌های اختصاصی برای یک ردیاب

ردیاب باید روی پورت مشخص شده در حال گوش کردن باشد. زمانی یک بسته‌ی HTTP دریافت می‌کند، باید آن را بررسی کند و پاسخ دهد.

announce را همیشه اول آدرس به شکل نمونه بگذارید، این آدرس محلی است که معمولا در ردیاب‌ها پاسخ را دریافت می‌کنند. محتوای بسته‌های درخواستی به ردیاب باید به این شکل باشند:

```
GET /announce?info_hash=<Info hash>&peer_id=<Peer's ID>&port=<Peer's port> HTTP/1.1
Host: <Tracker Address>
```

یک بسته‌ی درست HTTP را در متن ۲ می‌بینید.

```
GET /announce?info_hash=4de0&peer_id=salam&port=57847 HTTP/1.1
Host: 201.32.32.2
```

متن ۲، نمونه‌ی یک درخواست مورد نظر تمرین از سمت همتا به ردیاب

سپس باید ردیاب پایگاه‌داده خود در مورد همتاهای این تورنت را به روز کند و بسته‌ای شامل فهرست همتاهای این تورنت را ارسال کند. که هر خط آن به ترتیب شامل peer ID، IP و listening port است. مقدار Interval را همیشه 1000s بگذارید. بسته را به پورت ۸۰ همتا بفرستید.

```
HTTP/1.1 200 OK
Interval: 1000s
info hash: 01ca
hello 176.2.2.1 20311
salam 176.2.10.1 39881
```

متن ۳، نمونه‌ی محتوای پاسخ ردیاب به درخواست همتا

همیشه در پاسخ ردیاب خط اول و دوم را ثابت و مانند نمونه بگذارید. در پاسخ، اطلاعات همتای درخواست کننده را برنگردانید.

۳.۳. وظایف همتا

در custom information همتا در ابتدا عبارت `Peer Info:` آمده. در اولین خط Peer ID نوشته شده. سپس پورتی که روی آن گوش می‌دهد. سپس آدرس ردیاب(آدرس ردیاب همیشه IP آن است) همراه با پورتی که روی آن گوش می‌دهد آمده‌است. سپس تعداد پرونده‌های تورنت و بعد از آن اطلاعات مورد نیاز برای هر پرونده‌ی تورنت آمده است.

```
Peer Info:
salam
Listening port: 6881
201.32.32.2:31259
2
Info hash: 4de0
File size: 512
Piece size: 32
CRC divisor: 10011
hash: a2ec54ff02c98421
data: 9a4a892beef081e0658c3532719002d3fc26b73709606b326c...
Info hash: 01ca
File size: 256
Piece size: 32
CRC divisor: 11001
```

```
hash: ed119802
data: 1d4c8c75f0925891eb4cd1a005f4e120ffbdbfd3b1ddb70273...
The Mac Address: 14:32:F3:A2:10:00
```

متن ۴، نمونه‌ی داده اختصاصی یک همتا

اندازه پرونده‌ها و اندازه هر قطعه برای هر تورنت به بایت مشخص می‌شود. در تمرین به جای روش SHA-۱ از روش CRC برای تابع درهم ساز استفاده می‌کنیم. پس ابتدا مقسوم‌علیه CRC که همیشه یک عدد ۵ بیتی در مبنای ۲ است را می‌دهیم. پس کد درهم ساز همیشه یک عدد ۴ بیتی می‌شود که با یک رقم در مبنای ۱۶ قابل نمایش است. سپس مقدار درهم‌ساز قطعات را در مبنای ۱۶ می‌دهیم. به این صورت که هر نویسه[۱۴] کد درهم‌ساز یک قطعه است. پس طول عبارت hash به اندازه تعداد قطعات است. سپس اطلاعاتی که همتا از داده‌ی اصلی دارد میاید. توجه کنید که اطلاعاتی که همتا دارد به طول داده‌ی اصلی است ولی در بعضی قسمت‌ها اشتباه است. فرض اولیه شما باید این باشد که تمام داده‌های شما اشتباه است.

۱.۳.۳. فهرست دستورات همتا

> `check <peice number> of <torrent number>`

شما باید کد درهم‌ساز قطعه‌ی داده شده را بر حسب روش CRC و داده‌ی فعلی خود محاسبه کنید و با مقدار درهم‌ساز داده شده مقایسه کنید. اگر مساوی بودند، به این معنی است که داده شما در مورد این قطعه درست است و شما این قطعه را دارید (اگر فرض قبلی شما دارا نبودن قطعه بوده‌است باید تغییر کند). سپس عبارت `<peice number> data hash code matched` را چاپ کنید.

اگر مساوی نبودند باید عبارت `<peice number> data hash code did not match` را چاپ کنید.

همیشه منظور از `torrent number` شماره‌ی ترتیبی تورنت در custom information است که از یک شروع می‌شود.

> `check all of <torrent number>`

باید تمام کدهای درهم‌ساز قطعات برحسب داده‌های فعلی محاسبه شود، سپس کدهای درهم‌سازی که منطبق بود، باید فرض شود که آن قطعه را داریم. در انتهای انجام عملیات باید عبارت `we have <number of pieces> piece(s)` چاپ شود که <number of pieces> تعداد قطعاتی است که اکنون می‌دانیم داریم.

> `get peers of <torrent number>`

در این دستور باید شما به ردیاب درخواست HTTP بفرستید و اطلاعات فایل تورنت شماره <torrent number> را در آن قرار دهید. بعد از دریافت اطلاعات همتاها، آنها را ذخیره کنید. پورت مبدا را ۸۰ و مقصد را پورت ردیاب داده شده بگذارید.

[۱۴] Character

> `connect to <peer ID> from <torrent number>`

با فرستادن بسته‌ی handshake به همتای مقابل متصل می‌شوید. توجه کنید در این حالت هر دو در وضعیت choke و not interested هستید.

مقدار pstr را `BitTorrent protocol` بگذارید. در نتیجه pstrlen عدد ۱۹ یا `0x13` را می‌گیرد. مقدار info hash تنها ۴ بایت اول بخش info hash تورنت و مقدار peer ID هم با رشته‌ی شناسه‌ی همتا پر می‌شود و هر مقدار آن‌ها خالی ماند، با بایت صفر رها می‌شود.

شما باید در جواب بسته‌ی handshake، عبارت را چاپ کنید:

`bittorrent handshake packet received from <sender's peer ID>`

و بسته‌ی handshake را ارسال کنید. در هر حالتی که بسته‌ی handshake ارسال می‌کنید هم باید عبارت `bittorrent handshake packet sent to <destination's peer ID>` را چاپ کنید.

توجه کنید که info hash باید ۲۰ بایت باشد ولی در تمرین ما آن را ۴ کاراکتری در نظر گرفتیم. در بسته‌ها handshake پروتکل بیت‌تورنت شما همچنان ۲۰ بایت در نظر می‌گیرید ولی تنها ۴ بایت ابتدای آن را(بر اساس ASCII) پر می‌کنید. بقیه بایت‌ها باید صفر باشند.

> `unchoke to <peer ID> of <torrent number>`

اتصال خودتان را با این همتا به حالت unchoke دربیاورید. لازم است برای اعلام به همتای مقابل بسته‌ی مناسب ارسال کنید. همچنین در صورت دریافت بسته‌ی unchoke باید وضعیت همتا را به‌روزرسانی کرد و عبارت `<sender's peer ID> unchoked` را چاپ کرد.

> `choke to <peer ID> of <torrent number>`

به صورت مشابه حالت unchoke در این وضعیت هم باید بسته‌ی مناسب ارسال کرد و وضعیت را به روز کرد. در صورت دریافت این بسته هم باید عبارت `<sender's peer ID> choked` چاپ کنید.

`get interested to <peer ID> of <torrent number>` مشابه دو حالت قبل بسته‌ی مناسب و به روز رسانی وضعیت. در دریافت بسته عبارت `<sender's peer ID> is interested` را چاپ کنید.

> `get uninterested to <peer ID> of <torrent number>`

مشابه دو حالت قبل بسته‌ی مناسب و به روز رسانی وضعیت. در دریافت بسته عبارت `<sender's peer ID> is not interested` را چاپ کنید

>

`download piece <peice number> from <peer ID> of <torrent number>`

باید بسته‌ی مناسب برای درخواست قطعه را ارسال کنید. و عبارت `the piece requested` را چاپ کنید. در صورت not interest بودن نباید بسته‌ی ارسال کنید و عبارت `we are not interested` را چاپ کنید، در صورت choke بودن همتای مقابل، بسته را ارسال می‌کنید و همتای مقابل باید عبارت `we are choked to <peerID>` را چاپ کند و جوابی نفرستد.

- هر گاه دستوری وارد شد که peer ID آن را در فهرست تورنت مشخص شده نداشتید، این عبارت را چاپ کنید:

`unregistered peer id`

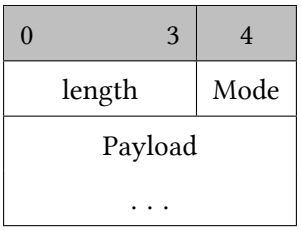
- هر گاه دستوری وارد شد که قبلا با peer ID آن اتصال برقرار نکردید این عبارت را چاپ کنید:

`we are not connected`

- هر جا که همتای جدید به لیست شما اضافه شد، چه در ردیاب، چه در یک همتا، عبارت زیر را چاپ کنید:

`new peer <PeerID> added to torrent <Torrent number>`

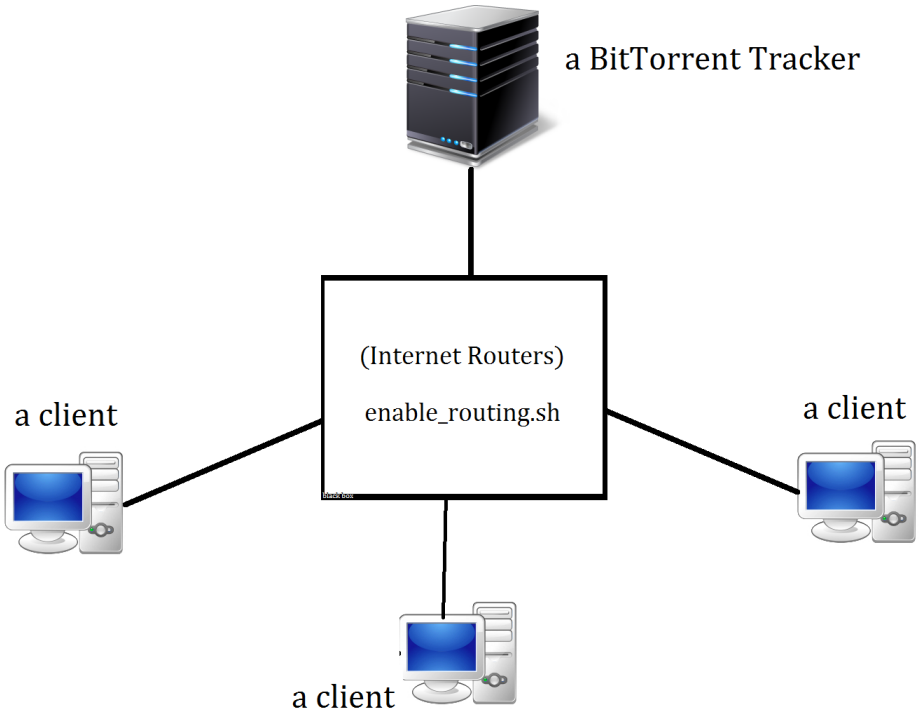
تمام بسته‌های پروتکل bittorrent به جز handshake به شکل زیر هستند:



برای توضیحات این پروتکل به بخش شرح پروتکل بیت‌تورنت مراجعه کنید.

۴. توپولوژی شبکه

نقشه‌ی تمرین، single_tracker_3_peers.map یک نقشه‌ی ساده با یک ردیاب و سه همتا است. در این نقشه ۳ فایل مجزای تورنت در شبکه قرار است به اشتراک گذاشته شود.



شکل ۳: توپولوژی برای آزمون

شما باید با اجرای دستور `new.sh` یک نسخه از نقشه را می‌سازید و یک jar که وظیفه‌ی ارسال بسته‌ها را دارد را اجرا می‌کنید. توجه کنید که در نقشه‌ای که برای نمره دهی قرار داده می‌شود تعداد همتاها و ردیاب‌ها ممکن است افزایش پیدا کنند.

۱.۴ نکات مهم

به بزرگ کوچکی حروف، ترتیب مقادیر و استاندارد مطرح شده در نمونه‌ها دقت کنید. در انتهای هر خط هیچ فاصله‌ای وجود ندارد. به نمونه‌ی رشته‌ی زیر که جواب یک ردیاب در قالب HTTP است دقت کنید(فرض شده جواب شامل یک همتا است):

```
"HTTP/1.1 200 OK\nInterval: 1000s\n" + Peer_ID + " " + IPtoStr(Peer_IP)
 + " " + str(listening_port)
```

همچنین تضمین می‌شود آدرس آی‌پی هر همتا منحصر به فرد باشد و در طول تست‌ها بین هر دو همتایی حداکثر یک ارتباط برقرار شود.

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 پوشه‌ی user-router را به همراه Makefile فشرده کرده (می‌توانید این کار را با اجرای دستور `make archive` انجام دهید) و از طریق وب‌سایت پرتو ارسال نمایید.