به نام خدا

شبکه‌های کامپیوتری

پروژه - شبکه Torrent

استاد

حسین پیوندی

تهیه و تدوین

تیم دستیاران درس - بخش پروژه

بهار 1403

موضوع پروژه : شبکه تورنت (Torrent)

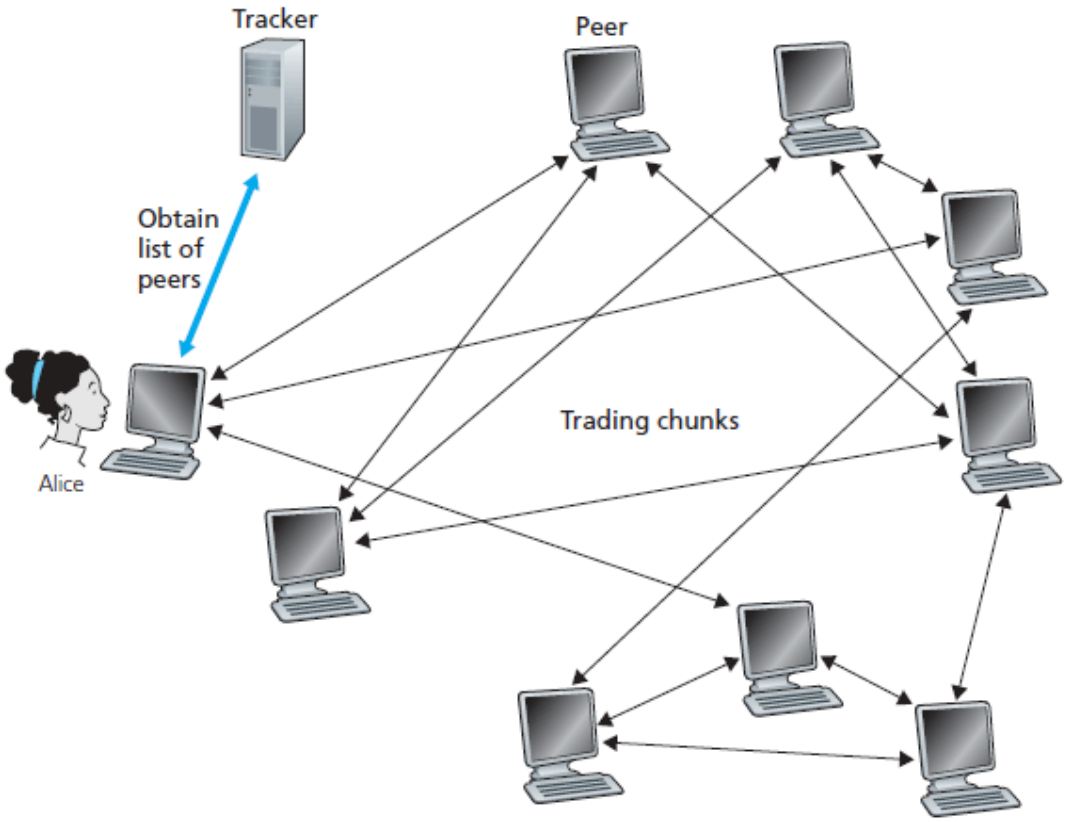
در این پروژه با استفاده از پروتکل های TCP و UDP شبکه ای مشابه با تورنت را پیاده سازی خواهید کرد.

تورنت چیست؟

تورنت یک پروتکل به اشتراک گذاری فایل به صورت peer-2-peer و غیرمتمرکز است. در این پروتکل به هر سیستمی که وارد شبکه شود و بخواهد فایلی دانلود کند یا آپلود کند، به طور کلی یک peer گفته میشود.

همچنین سرورهای متمرکزی به اسم tracker وجود دارند که کار آنها نگه داشتن یک سری اطلاعات است. مثلا این اطلاعات، شامل این است که هر peer در شبکه، چه قسمتهایی از کدام فایلها را دارد.

شکل زیر یک شمای کلی از نحوه کارکرد این شبکه را نشان میدهد.



به صورت خلاصه این شبکه بدین صورت کار میکند که زمانی که یک peer، فایلی را از بقیه دانلود میکند، خودش تبدیل به یک آپلودکننده (seeder) برای آن فایل میشود و میتواند همان فایل را با بقیه به اشتراک بگذارد. بدین صورت تعداد seederهای یک فایل به صورت غیر متمرکز زیاد میشود.
برای درک نحوهی کارکرد تورنت فرض کنید که میخواهیم فایلی را با بقیه در شبکهی تورنت به اشتراک بگذاریم. فرض کنید که کامپیوتر A میخواهد فایل F را در شبکه ی تورنت به اشتراک بگذارد. در ابتدا A فایل F را به بخشهای دو مگابایتی میشکند؛ به هر کدام از این بخشها، یک chunk گفته میشود. سپس به tracker اعلام میکند که تمامی chunk های فایل F را در اختیار دارد و اگر کسی این فایل را درخواست کرد، tracker او را به A وصل کند. در واقعیت فایلها به کمک checksum آنها از هم متمایز می شوند ولی در این پروژه برای سادگی کار بر اساس نام فایل، فایلها را از هم تفکیک کنید!

حال فرض کنید که کامپیوتر B بخواهد فایل F را دانلود کند. در ابتدا B از tracker میخواهد که لیست کامپیوترها و chunkهایی از فایل که هر کدام از کامپیوترها دارند را برای او ارسال کند. با این کار tracker مشخصات کامپیوتر A را برای B میفرستد و همچنین مشخص میکند که A تمام chunkهای فایل را دارد. سپس B مستقیما به A درخواستی حاوی پیام «یکی از chunkهای فایل را برای من ارسال کن» میفرستد. بعد از دریافت کامل chunk، کامپیوتر B به tracker اعلام میکند که حال من نیز این chunk فایل را دارم. با این کار در صورتی که یک کامپیوتر دیگر مثلا C از tracker درخواست این chunk از فایل را بدهد ،tracker به او اعلام میکند که کامپیوترهای A و B این chunk را در اختیار دارند. با این کار کلاینتهای جدید، به مرور انتخابهای بیشتری برای دانلود فایل دارند.

صورت پروژه:

در این تمرین، ما قصد داریم که شبکه ی تورنت را به صورت ساده شده پیاده سازی کنیم. در این شبکه لازم است که حداقل سه peer و حداقل یک tracker داشته باشیم. (در صورت تمایل میتوانید تعداد peerها و trackerها را بیشتر نیز کنید اما این کار اجباری نیست و صرفا حداقل تعداد گفته شده باید رعایت شده باشد). تنها tracker میداند هر فایل در کدام peerها دانلود شده به صورت کامل قرار دارند و هنگامی که یک فایل به خصوص درخواست شود، باید اطلاعاتی همچون سایز فایل و لیست peerهایی که فایل را به صورت کامل دارند برای درخواست کننده فرستاده شود. سپس درخواست

کننده، با توجه به جواب tracker یکی از peerهایی که آن فایل را در اختیار دارد به صورت تصادفی انتخاب میکند و درخواست دانلود برای او ارسال میکند. در انتها peer درخواست دهنده، بعد از دانلود فایل، به حالت seed میرود. یعنی برنامه همچنان باز میماند و در صورت نیاز ،بقیهی peerها میتوانند از او درخواست فایل را بکنند. همچنین ،tracker باید لاگ فایلهای آپلود و دانلود شده را به همراه کسی که آن را آپلود/دانلود کرده است، داشته باشد.

برنامه هایی که باید پیاده سازی کنید:

برنامه tracker:

این برنامه همان طور که گفته شد وظیفه ی نگهداری اطلاعات فایلها و peerها را دارد. این برنامه باید قبل از اجرا یک ورودی به صورت IP:PORT به عنوان مثال 127.0.0.1:6771 دریافت کند. اجرای برنامه به صورت فوق باید باعث شود که tracker بر روی پورت UDP 6771 گوش بایستد و درخواستها را پاسخ دهد. (به صورت local)

برنامه peer:

ابتدا توجه کنید که برای تمایز peerها از یک دیگر، هر peer باید یک اسم یا مثلا id داشته باشد. در دنیای واقعی، کلاینتهای BitTorrent امکان دانلود کردن و به اشتراک گذاری چندین فایل را به صورت همزمان میدهند. اما برای سادگی در این تمرین فرض میکنیم که برنامهی peer تنها در یکی از دو حالت share یا get شروع به کار میکند.

حالت share:

در این حالت می خواهیم که از روی کامپیوتر خود فایلی را در شبکه ی تورنت به اشتراک بگذاریم. همان طور که گفته شد فایلها به اسمشان در شبکه شناخته میشوند. فرض کنید که فایلی با اسم تکراری به شبکه داده نمیشود. یک نمونه از آرگومانهای پیشنهادی برنامه به صورت زیر میباشد:

share <FILENAME> <TRACKER_ADDRESS> <LISTEN_ADDRESS>

به عنوان مثال:

share myfile.txt 127.0.0.1:6771 127.0.0.1:52611

فایل myfile.txt را در شبکه ی تورنت که tracker آن در آدرس 127.0.0.1:6771 قرار دارد به اشتراک میگذارد. همچنین در صورتی که کسی بخواهد این فایل را دریافت کند میتواند از آدرس 127.0.0.1:52611 به این peer متصل شود و درخواست فایل را بکند.

حالت get:

در این حالت ما میخواهیم که فایلی را از شبکه دریافت کنیم. در ابتدا باید peer ما از tracker بخواهد که اطلاعات فایل را برایش بفرستد. سپس یکی از peerها که این فایل را دارد را به صورت تصادفی (random) انتخاب کند و درخواست فایل را از او بکند.

نحوه آرگومان دادن در این حالت دقیقا مثل حالت قبل است، صرفا به جای share از get استفاده میکنیم و بقیه چیز ها مشابه هستند. به عنوان مثال:

get myfile.txt 127.0.0.1:6771 127.0.0.1:52612

با استفاده از این دستور ما از tracker با آدرس 127.0.0.1:6771 میخواهیم که اطلاعات فایل myfile.txt را برای ما بفرستد. نکته ای که باید در اینجا دقت کنید این است که برخلاف یک دانلود عادی از سایت زمانی که دانلود تمام میشود برنامه نباید تمام شود؛ بلکه تبدیل به یک seeder شود. بدین معنا که چیزی دانلود نمیشود بلکه صرفا آپلود صورت میگیرد.

لاگ سیستم:

با هر درخواستی که از یک peer برای tracker ارسال میشود، نام آن peer، درخواست آن ،peerهایی که آن فایل را در اختیار دارند و در نهایت موفق بودن یا نبودن در گرفتن فایل در tracker ثبت میشود که با زدن دستور request logs در خط فرمان tracker نشان داده میشوند. همچنین هر فایلی که در شبکه منتشر میشود یک لاگ برای tracker ثبت میکند که در صورت زدن دستور all-logs در خط فرمان tracker، کلیه این لاگها نمایش داده میشود (که هر قسمت از کدام فایل در دست کدام peerها است) و در صورت زدن <file_name> file_logs در خط فرمان tracker، لاگهای مربوط به یک فایل نمایش داده میشوند که در صورت عدم وجود فایل، باید یک پیام خطای مناسب نمایش داده شود. در برنامه peer نیز باید لاگ تمامی پاسخ های آمده از طرف سرور جهت گرفتن یک قسمت از فایل ثبت شود که با دستور request logs در خط فرمان peer نمایش داده میشوند. هنگام وصل شدن هر peer به tracker نیز، یک لاگ اتصال حاوی نام یا آیدی peer در برنامه tracker نمایش داده میشود. هنگام اتصال و قطع شدن peer از tracker نیز باید لیست فایلهای مربوط به آن peer اپدیت شود و نیز یک لاگ قطع شدن حاوی نام یا آیدی peer در برنامه tracker نمایش داده میشود.

نکات پیاده سازی:

- بین tracker و هر peer حتما از یک سازوکار keep alive یا ping-pong یا heartbeat استفاده کنید. این سازوکار به tracker کمک میکند تا متوجه شود کدام peer ها هنوز در شبکه هستند. فرض کنید که یکی از peer ها که حاوی فایلی باشد، از شبکه خارج شود. در این صورت اگر peer دیگری درخواست آن فایل را برای tracker ارسال کند، چون از peer حاوی آن فایل از شبکه خارج شده نمیتواند به این درخواست پاسخ دهد و peer درخواست دهنده معطل میماند! (این سازوکار میتواند به سادگی صرفا مشابه فرستادن یک بسته خاص در بازههای زمانی مشخص برای tracker باشد. در صورتی که بعد از مدتی معین ،tracker به نتیجه رسید که آن peer از شبکه خارج شده، باید اسم او را از لیست فایلهایی که آن peer دارنده آنها بوده، پاک کند، تا دیگر آن را به کسی ندهد.
- ارتباط بین tracker و peer باید به صورت UDP و ارتباط بین peerها باید به صورت TCP باشد.
- پروتکل ارتباطی بین تمام برنامه ها به عهده ی خودتان است.
- لازم نیست که برخلاف شبکه ی واقعی تورنت فایلها را chunk chunk کنید.
- دقت کنید که فایلها نباید از طریق tracker دانلود شوند! بلکه tracker صرفا باید آدرس IP و شماره پورت peerهایی را برگرداند که فایل درخواست شده را دارند.
- تمام برنامه ها باید به صورت multi threaded پیاده سازی شوند. به عنوان مثال tracker میتواند همزمان جواب چندین نفر را دهد یا اینکه فایل میتواند برای همزمان برای چند نفر آپلود شود.
- منظور از خط فرمان (command prompt) همان محیط کنسول در IDE است. در واقع خود tracker یا peer، یک shell ساده تعریف میکنند.
- دقت کنید که برنامه tracker و برنامه peer، باید مستقل از هم پیاده سازی شده باشند و هرکدام را جداگانه اجرا کنیم.
- شما فقط مجاز به استفاده از زبان های برنامه نویسی جاوا یا پایتون هستید.

امتیازی:

در صورتی که فایل هارا مثل شبکه اصلی تورنت به صورت chunk chunk ارسال کنید، بسته به درستی پیاده سازی به اندازه 10% نمره کل پروژه براتون امتیازی لحاظ خواهد شد.

موفق باشید!