به نام خداوند بخشنده‌ی مهربان

تمرین برنامه‌نویسی سوم[1]
شبکه‌های کامپیوتری

مدرس: مهدی خرازی **بهار ۱۳۹۰**

در این تمرین شما با گسترش برنامه‌ای که در مراحل قبل نوشتید، یک **سیستم به اشتراک‌گذاری فایل به صورت توزیع شده**[2] را **طراحی** و **پیاده‌سازی** خواهید کرد.

مقدمه

در مرحله قبل شما یک سیستم به اشتراک گذاری فایل را پیاده سازی کردید. در این سیستم وجود یک سرور مرکزی لازم بود تا اطلاعات مربوط به هر فایل-شامل آدرس و پورت دارنده آن فایل- را در اختیار کاربران قرار دهد. اولین ایرادی که می‌توان به این مکانیزم وارد کرد این است که با از کار افتادن سرور، سیستم قابل استفاده نخواهد بود و به عبارتی، کار کردن سیستم وابسته به یک گره در شبکه خواهد بود.

در این تمرین شما پروتکلی را طراحی و پیاده‌سازی می‌کنید که امکان حذف سرور از شبکه را فراهم کند. به عبارت دیگر لازم است که برنامه شما طوری عمل کند که اطلاعات مربوط به فایل‌ها، بین گره‌های موجود در شبکه به طور یکنواخت توزیع شوند. همچنین باید مکانیزمی بیاندیشید که با ورود یک گره به شبکه، آن گره وظیفه نگهداری بخشی از اطلاعات را برعهده بگیرد و از بار موجود روی سایر گره‌ها بکاهد. به طور مشابه زمانی که یک گره قصد خروج از شبکه را دارد، باید اطلاعاتی که مسئولیت آنها را دارد به سایر گره‌های موجود در شبکه منتقل کند تا این اطلاعات حفظ شوند.

پروتکل Chord، پروتکلی برای سیستمهای P2P DHT است که این امکانات را در اختیار شما قرار می‌دهد. البته شما می‌توانید از هر پروتکلی که میخواهید استفاده کنید، تنها به یاد داشته باشید که برنامه شما باید طوری عمل کند که نیازمندی‌های گفته شده در بالا را برآورده سازد.

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

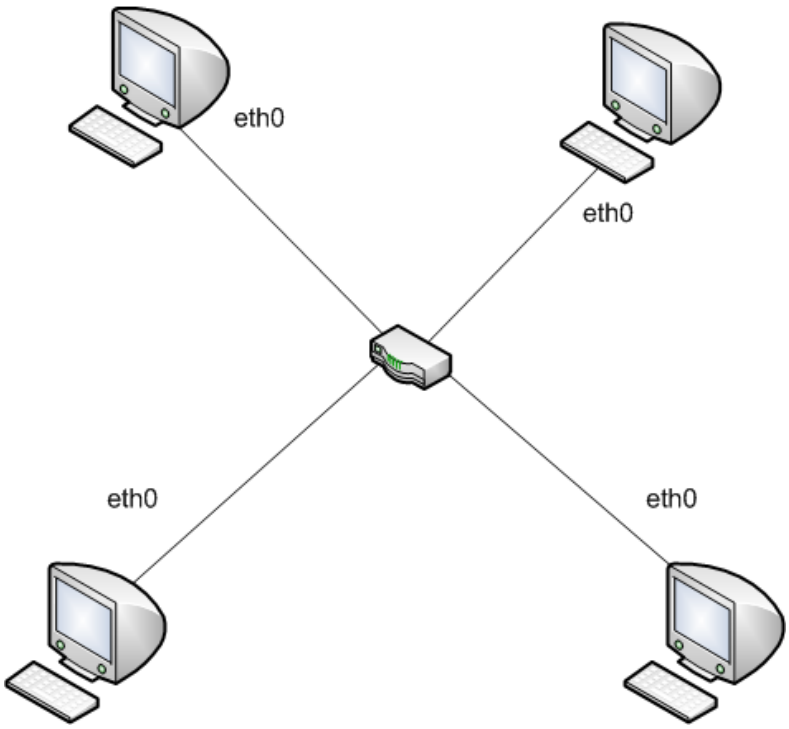
[2] Distributed File Sharing System

محیط

در این تمرین نیز، مانند تمرین قبل، از محیط پرتو[3] برای اجرای برنامه‌ی شما استفاده خواهد شد. برای این منظور لازم است برنامه‌های خود را بر روی «چارچوب کاربر»[4] پرتو پیاده‌سازی کنید. برای اطلاعات بیشتر در مورد این چارچوب به مستند «راهنمای چارچوب کاربر» مراجعه کنید.

توپولوژی

توپولوژی اولیه‌ی شبکه که در اختیار هر دانشجو قرار می‌گیرد مانند شکل ۱ است. البته برنامه‌ی شما به هیچ وجه نباید وابسته به توپولوژی باشد و یا فرض خاصی در مورد توپولوژی انجام دهد.



شکل ۱- توپولوژی اولیه

هر گره در داخل شبکه از طریق interface شماره صفر خود به روتر داخلی متصل است. این روتر داخلی وظیفه مسیریابی بین گره‌ها را بر عهده دارد.

[3] Portable And Reliable Tool fOr Virtualization (PARTOV)
[4] Client Framework (CF)

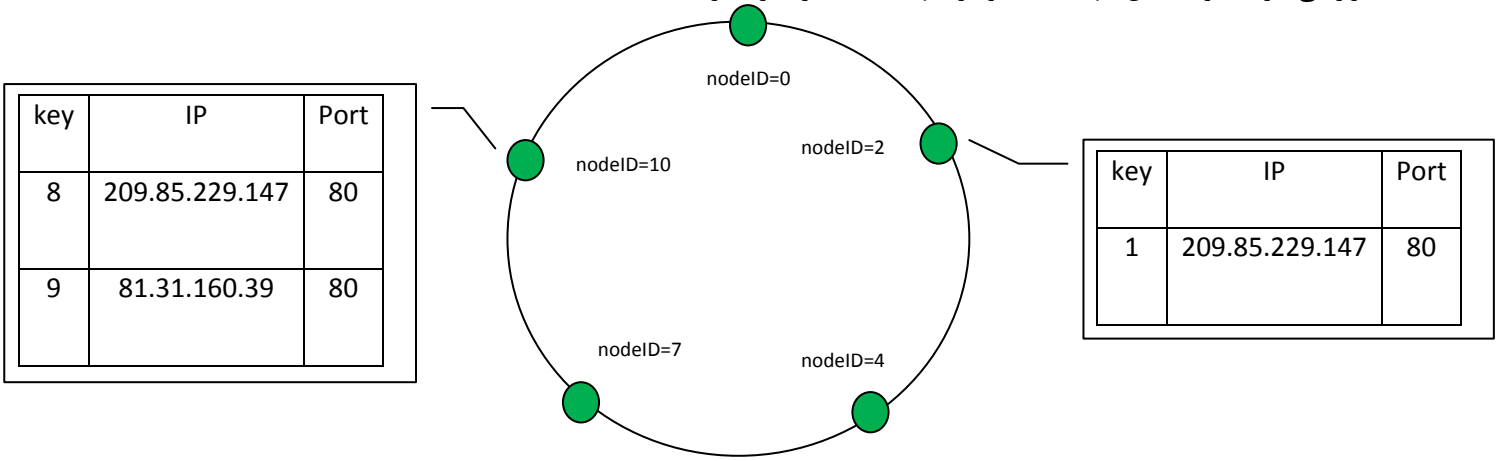
انتظارات

شما باید تنها برنامه مربوط به گره‌ها را بنویسید. Makefile را طوری تغییر دهید که با اجرای فرمان make فایل اجرایی با نام node.out تولید شود. این برنامه همان برنامه‌ای است که باید بر روی گره‌ها اجرا شود و باید آن را با پارامترهایی مناسب اجرا کنید تا بر روی device مناسب در توپولوژی قرار گیرد. نحوه‌ی اجرا کردن این برنامه با پارامترهای مناسب در فایل‌هایی به همراه «چارچوب کاربر» به شما داده شده است.

توضیح اجمالی پروتکل

پروتکل Chord یک پروتکل DHT است. در این پروتکل برای هر گره یک شناسه منحصر به فرد در نظر می‌گیریم که آن را NodeID می‌نامیم[۵]. همچنین هر گره اطلاعات مربوط به تعدادی فایل را در خود نگهداری می‌کند. این اطلاعات به صورت زوج مرتب (Key,Value) هستند که برای به دست آوردن مقدار Key متناظر با هر فایل باید نام آن را با استفاده از یک تابع درهم-ساز[۶]، به یک عدد hash کنیم.

مطابق با شکل ۲، گره‌ها را در این شبکه به ترتیب مقدارNodeID آنها روی یک حلقه می‌چینیم. هر گره، اشاره‌گری به گره ماقبل و مابعد خود در حلقه خواهد داشت که با استفاده از آن می‌تواند، اطلاعات مورد نظر خود را به دست بیاورد. همچنین هرگره مسئول نگهداری اطلاعات مربوط به فایل‌هایی خواهد بود که Key مربوط به آنها، بین NodeID آن گره و NodeID گره قبلی آن قرار می‌گیرند. (برای مثال جهت ساعتگرد را جهت مثبت در نظر بگیرید)



شکل ۲- چیدمان گره‌ها در پروتکل

[۵] این شناسه را می‌توان با hash کردن آدرس IP هر گره به دست آورد. لزومی نیست که حتما خروجی تابع درهم‌سازی شما یک عدد باشد. کافی است بتوانید مقادیر را با یکدیگر مقایسه کنید.

[۶] برای تضمین یکنواخت بودن نسبی توزیع فایلها می‌توانید از الگوریتم SHA-1 برای درهم‌سازی استفاده کنید.کتابخانه `openssl/sha.h` توابع مربوط به این الگوریتم را در اختیار شما قرار می دهد. می‌توانید برای اطلاع بیشتر به اینجا مراجعه کنید.

هنگامی که درخواست دریافت اطلاعات مربوط به یک فایل به یک گره وارد میشود، آن گره چک می‌کند که آیا خود مسئول آن فایل است یا اینکه باید درخواست را برای گره بعدی ارسال کند. به این ترتیب درخواست آنقدر میان گره‌ها رد و بدل می‌شود تا اطلاعات آن یافت شده یا ثابت شود که در شبکه وجود ندارد[۷].

هنگام ورود یک گره به شبکه، آن گره باید با استفاده از پیامهایی که با سایر گره‌های موجود در شبکه رد و بدل می‌کند، جایگاه خود بر روی حلقه را بیابد و پس از آن اطلاعاتی که مربوط به اوست را از همسایه خود دریافت کند.

هنگام خروج هم، گره مورد نظر لازم است که اطلاعات را در اختیار همسایه‌اش قرار دهد تا داده‌ای از دست نرود.

برای آشنایی بیشتر، با پروتکل Chord توصیه می‌شود به مراجع زیر مراجعه کنید:

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

http://en.wikipedia.org/wiki/Chord_%28peer-to-peer%29

برنامه مربوط به گره‌ها

برنامه مربوط به هر گره باید از ۵ دستور زیر که در ادامه توضیحات مربوط به آن‌ها خواهد آمد، پشتیبانی کند.

۱ - دستور join:

با وارد کردن این دستور در ترمینال مربوط به یک گره، آن گره باید وارد شبکه گره‌های درگیر در پروتکل شود. برای این منظور لازم است که این گره، آدرس گره دیگری که قبلا وارد شبکه شده است را داشته باشد. فرض کنید که در custom information هرگره، لیستی از آدرس گره‌هایی که احتمالا قبل از این گره وارد شبکه شده‌اند، ذکر شده باشد[۸]. بنابراین با وارد کردن این دستور، گره سعی می‌کند که با کمک این آدرس‌ها، گره دیگری در شبکه را یافته و از طریق آن، جایگاه خود بر روی حلقه را بیابد. مشخص است که اگر پاسخی از گره‌های ذکر شده، دریافت نشود این گره اولین گره‌ای است که وارد شبکه می‌شود. مدت زمان مهلت برای ارسال پاسخ برای هر گره داخل لیست را ۵ ثانیه در نظر بگیرید.

۲ - دستور put:

این دستور برای ذخیره کردن اطلاعات مربوط به یک فایل در شبکه استفاده می‌شود. بعد از وارد کردن این دستور باید در خطوط بعد، به ترتیب نام فایل، آدرس IP مربوط به دارنده فایل و شماره Port متناظر ذکر شود. برای مثال می‌توان روی یکی از گره‌ها دستور زیر را اجرا کرد:

[۷] برای این تمرین نیازی به پیاده‌سازی finger table نیست.

[۸] توجه کنید که نمی توانید فرض کنید که این لیست شامل همه گره‌های موجود در شبکه است.

put
logo.jpg
1.2.3.4
8080

هنگامی که این دستور بر روی گره‌ای اجرا می‌شود، لازم است که key متناظر با filename تولید شده و مقادیر آدرس و پورت بر روی گره‌ای که مسئولیت آن مقدار key را دارند، ذخیره شود. برای این منظور لازم است مکانیزمی طراحی کنید که بتوانید گره مسئول هر Key را بیابید.

۳ - دستور get:

از این دستور برایی دریافت اطلاعات یک فایل استفاده می‌شود. بعد از وارد کردن این دستور باید در خط بعد نام فایل مورد نظر آورده شود، سپس جستجو برای پیدا کردن آدرس IP و شماره Port سرور دارای این فایل صورت گرفته و در خروجی نوشته می‌شود. برای مثال ورودی و خروجی زیر را در نظر بگیرید:

Input:
get
logo.jpg
Output:
1.2.3.4
8080

OR

Input:
get
index.html

Output:

file not found!

۴ - leave
:

هنگامی‌که این دستور وارد می‌شود، گره مورد نظر قصد خروج از شبکه را دارد. بنابراین باید، اطلاعات خود را به نحوی به گره‌ای که هنوز در شبکه حضور دارد (در صورت وجود چنین گره‌ای) منتقل کند.

۵ - print:

با وارد کردن این دستور باید اطلاعات مربوط به گره را به صورت زیر در خروجی استاندارد چاپ کنید. اطلاعات داخل table در واقع اطلاعات مربوط به فایلهایی است که این گره مسئول نگهداری از آنهاست. توجه کنید که اطلاعات هر فایل دقیقا توسط یک گره نگهداری می‌شود.

successor-ip= 192.168.6.2
predecessor-ip= 192.168.6.2
table:
filename1 ip1 port1
filename2 ip2 port2

یکسان بودن آدرس IP همسایه‌ها به این معنی است که تنها ۲ گره در شبکه حضور داشته‌اند. همچنین اگر گره‌ای به تنهایی در شبکه بود و همسایه‌ای نداشت، آدرس IP برای successor و predecessor آن به صورت 0.0.0.0 چاپ خواهد شد.

همچنین قالب custom information برای گره‌ها به صورت زیر خواهد بود:

Mac-of-the-Router

Candidate1-IP-Address

Candidate2-IP-Address

...

نکات ضروری

اولا توجه کنید که هدف از این الگوریتم، توزیع یکنواخت داده‌ها میان گره‌هاست. بنابراین به هیچ وجه مجاز نخواهید بود که همه اطلاعات را در همه گره‌ها ذخیره کنید. همچنین هر گره تنها اجازه دارد که آدرس IP مربوط به دو همسایه خود را ذخیره کند و نمی‌توانید از بسته‌های broadcast برای انتقال پیام‌ها استفاده کنید.

ثانیا برای تست کردن برنامه خود می‌توانید سناریو زیر را در نظر بگیرید:

۱ - شبکه اولیه را با حضور ۲ تا از گره‌ها ایجاد کنید.
۲ - اطلاعات مربوط به تعدادی فایل را توسط گره‌های مختلف وارد شبکه کنید. برای مثال می‌توانید از دستوراتی مانند دستور زیر استفاده کنید:

put

/intl/en_com/images/srpr/logo1w.png

80

۳ - درخواست دریافت اطلاعات فایل را به یکی از گره‌ها بدهید.

get

/intl/en_com/images/srpr/logo1w.png

یابد خروجی زیر را مشاهده کنید:

209.85.229.147

80

۴ - دو گره دیگر را وارد شبکه کنید.

۵ - دو گره اولیه را از شبکه خارج کنید.

۶ - بار دیگر درخواست دریافت اطلاعات فایل را ارسال کنید.

در این مورد هم باید بتوانید اطلاعات را به درستی دریافت کنید.

ثالثا برای سادگی بیشتر می‌توانید فرض کنید که هیچ دو گره‌ای همزمان برای اتصال به شبکه اقدام نمی‌کنند. همچنین بدیهی است که گره‌ای که خود هنوز وارد شبکه نشده است(دستور join برای آن وارد نشده است) نمیتواند پاسخ گره‌های دیگری که درخواست ورود به شبکه دارند را بدهد و این بسته‌ها را نادیده می‌گیرد.

رابعا توجه کنید که در این مرحله شما مکانیزمی را پیاده سازی کرده‌اید که به کمک آن می‌توانید اطلاعات مربوط به فایلها را به صورت توزیع شده ذخیره و بازیابی کنید. با اتصال این قسمت به برنامه‌ای که در فاز قبل نوشتید می‌توانید قابلیت دانلود فایلها را هم به گره‌های خود بدهید. کافی است با استفاده از دستور put اطلاعات مربوط به فایلی که در صورت فاز قبل به شما داده شده را در شبکه ذخیره کنید. سپس با استفاده از دستور get مقادیر IP و Port دارنده آن را به دست آورده و از برنامه client که در مرحله قبل نوشتید استفاده کرده (با فراهم آوردن host مربوطه)و فایل را دانلود کنید. در هر صورت برای گرفتن نمره این مرحله **نیازی به اضافه کردن قابلیت دانلود به گره‌ها نیست**.

و در نهایت

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 پوشه های اضافی را در بر گیرد. تنها پوشه‌ی user و فایل Makefile و نه پوشه‌ی دربرگیرنده‌ی آن‌ها را zip کرده و ارسال نمایید.