گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

در ابتدا طرز کار و ساختار برنامه همراه با اجراهای نمونه گفته شده و سپس بخش‌های کد گفته شده.

معماری کلی برنامه

در این شبیه‌سازی ۳ فایل اجرایی مختلف ساخته می‌شود که به ترتیب برای System ،Interface و Switch هستند. کاربر از طریق Interface دستورات را وارد می‌کند و بر اساس دستور ساخت Switch یا System، یک پردازه جدید با استفاده از سیستم‌کال‌های fork و exec ساخته شود و فایل‌اجرایی مرتبط با آن‌را اجرا می‌کند. پردازه Interface از طریق unnamed pipe با پردازه‌های فرزندانش (switch و systemها) ارتباط دارد و دستورات مرتبط را به آن‌ها منتقل می‌کند. رابطه سوییچ‌ها و سیستم‌ها نیز از طریق named pipe صورت می‌گیرد. به این صورت که هر سوییچ برای هر پورتش دو named pipe با نامی به فرمت زیر می‌سازد:

```
s-[id]-[port]-[in/out]
```

برای مثال سوییچی با شناسه ۳ برای فریم‌هایی که از طریق پورت ۱ خود دریافت می‌کنم پایپی به نام s-3-1-in می‌سازد. و برای فریم‌هایی که پورت ۱ می‌فرستد پایپ s-3-1-out را می‌سازد. برای اینکه مشکل در خواندن و نوشتن هم‌زمان پیش نیاید از دو پایپ به صورت یک طرفه استفاده شده.

سوییچ‌ها و سیستم‌ها با استفاده از سیستم‌کال select همزمان آماده دریافت پیام از پایپ پورت‌ها و پایپ دستور هستند.

همینطور سیستم‌ها هنگام دریافت فایل، یک directory با نام systemN می‌سازند و فایل را در آن‌جا ذخیره می‌کنند.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

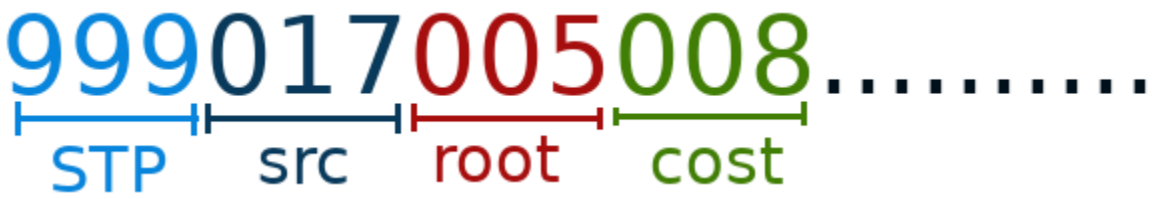
ساختار فریم

در این شبیه‌سازی از آرایه char* به عنوان فریم استفاده شده و همیشه با سایزی مشخص (۱۰۲۴ اینجا تعریف شده) ارسال و خوانده می‌شود. در فریم‌ها ۱۰ خانه ابتدای آرایه نقش هدر را دارد که نشان می‌دهد مقصد و مبدا فریم کدام سیستم‌ها هستند و سایز اطلاعاتی که ادامه فریم آمده چقدر است. برای مثال در تصویر بالا یعنی سیستم با شناسه ۳، برای سیستم ۱۰، فریمی با سایز ۵۴۳ فرستاده.

در فریم درخواست فایل جای سایز عدد ۹۹۹۹ قزار می‌گیرد و سپس اسم فایل درخواستی می‌آید.

برای ارسال فایل ابتدا یک فریم با اسم فایل و سایز فایل فرستاده می‌شود و سپس خود فایل در چندین فریم فرستاده می‌شود و سیستم گیرنده براساس سایز فایل که در فریم اول دریافت کرده تکه‌های فایل‌را دریافت میکند در یک فایل جدید می‌نویسد.

فریم STP



فریم STP نیز به شکل تصویر است و اطلاعات مورد نیاز اجرای پروتکل را دارد. برای مشخص کردن این پیام، همیشه ۳ عنصر اول آن 999 است.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

دستورات ورودی

```
add_switch <id> <portCount>
add_system <id>
connect <system_id> <switch_id> <port_number>
send <src system> <dst system> <file name>
recv <src system> <dst system> <file name>
connect_switch <s1_id> <s1_port> <s2_id> <s2_port>
run_stp
```

الگوریتم STP

ابتدا باید یک سوییچ به عنوان ریشه درخت (روت) در نظر گرفته شود و همه سوییچ‌ها پورتی که به روت راه دارد را پیدا کنند و پورت‌های دیگر خود را در صورتی که در آن اتصال سوئیچ designated نیستند، در حالت blocking قرار دهند.

در ابتدای اجرای الگوریتم همه سوییچ‌ها فکر می‌کنند که ریشه هستند. و پیام stpای می‌فرستند که در آن خود را ریشه معرفی می‌کنند و فاصله را صفر می‌گویند.

هر سوییچ زمانی که یک پیام STP دریافت می‌کند، ابتدا بررسی می‌کند که اطلاعات آن بهتر از اطلاعات خودش است یا نه. معیار بهتر بودن به ترتیب اولویت شامل موارد زیر است:

- داشتن روتی با id کوچک‌تر
- کمتر بودن فاصله تا روت
- داشتن id کوچک‌تری نسبت به فرستنده‌ی اطلاعات ثبت شده فعلی

اگر بهتر باشد اطلاعات خودش را با این پیام آپدیت می‌کند و از این به بعد این پیام را برای بقیه ارسال می‌کند. (فقط هزینه را ۱+ می‌کند.)

ولی اگر پیام جدید مسیر روت بهتری را معرفی نکند، سوییچ بررسی می‌کند که آیا در آن اتصال designated است و وضعیت بهتری نسبت به سوییچ دیگر دارد یا خیر. یعنی فاصله‌اش نسبت به روت کمتر است یا id کمتری داشته باشد. اگر بفهمد سوییچ

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

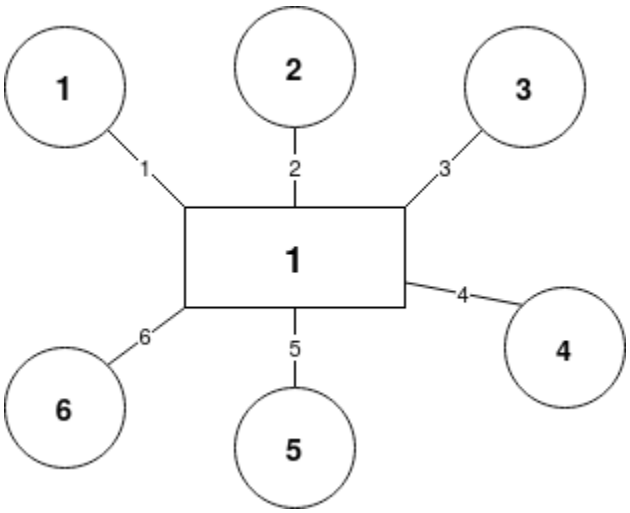
designated است کاری نمی‌کند و باز هم به آن پورت پیام ارسال می‌کند. ولی اگر متوجه شود که designated نیست، آن پورت را در حالت blocking قرار می‌دهد. به این ترتیب یال‌هایی حذف می‌شوند و دور از بین می‌رود.

نحوه اجرای برنامه

```
> make clean
> make all
> ./sim.out
```

اجراهای نمونه

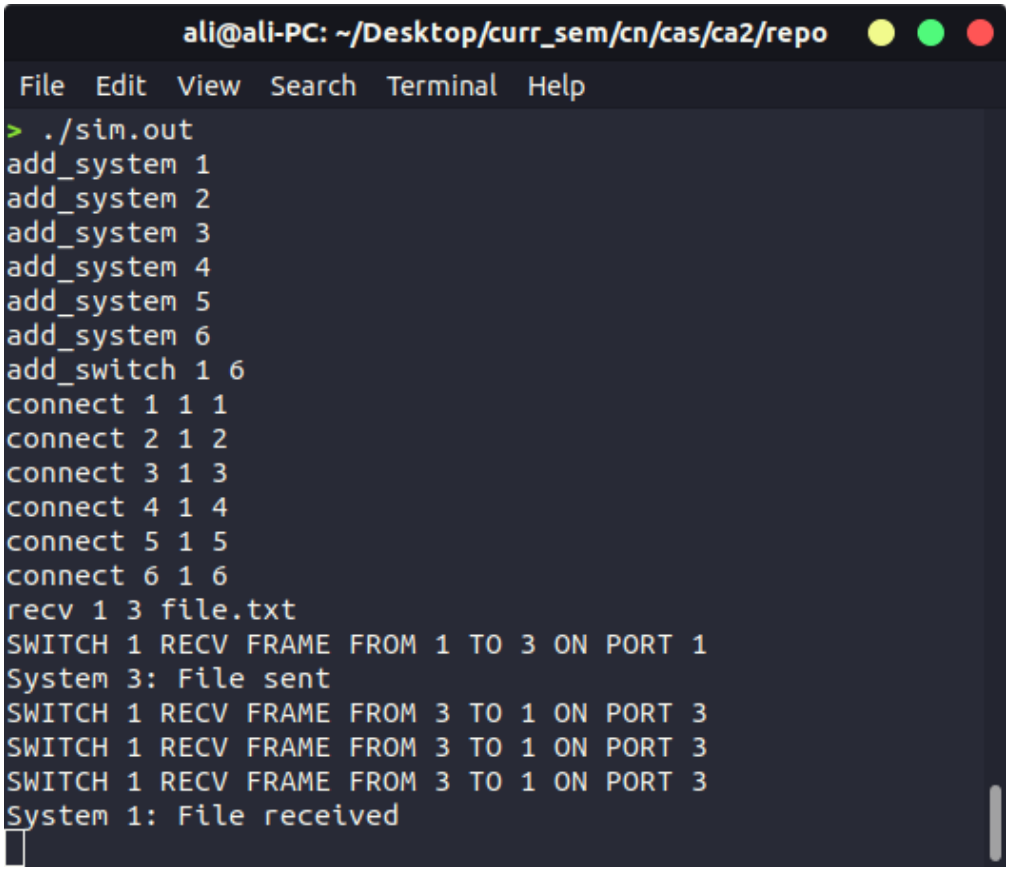
توپولوژی گام ۳



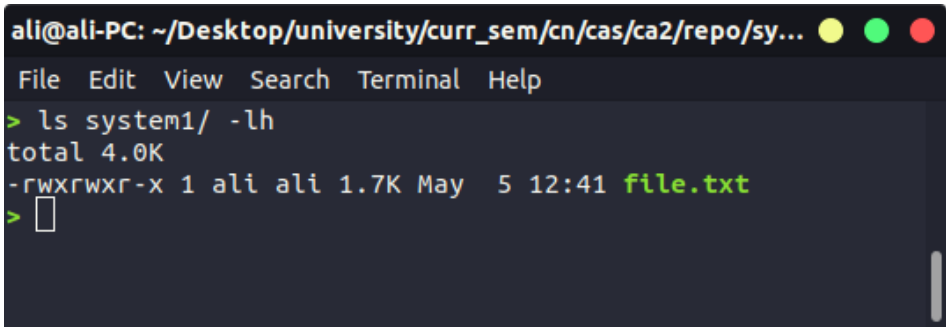
گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626



همینطور که دیده می‌شود سوییچ ابتدا یک فریم از سیستم ۱ به سیستم ۳ دریافت می‌کند که همان فریم درخواست فایل است. سپس ۳ فریم از سمت سیستم ۳ به سیستم ۱ دریافت کرده که فریم اول مشخصات فایل است و دو فریم بعدی خود فایل هستند.

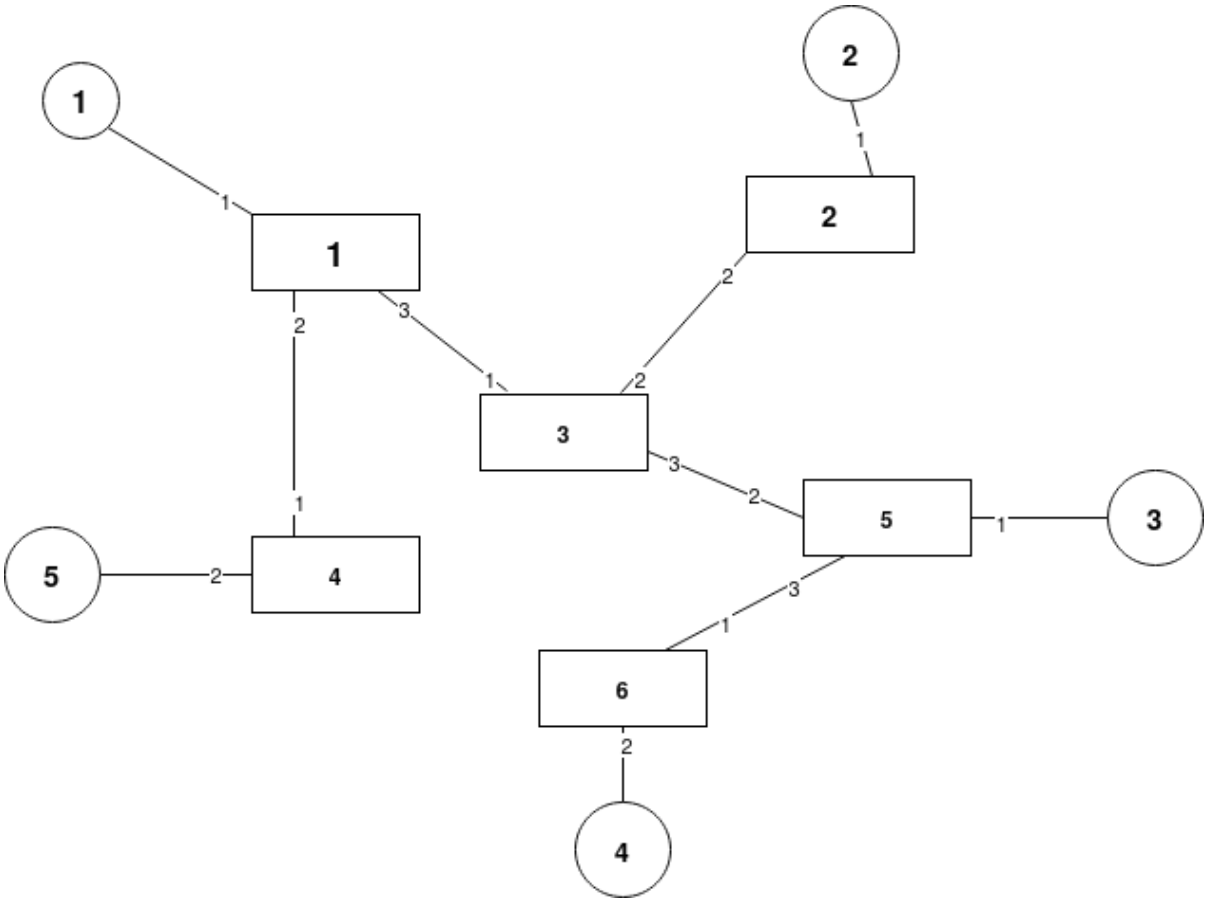


گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

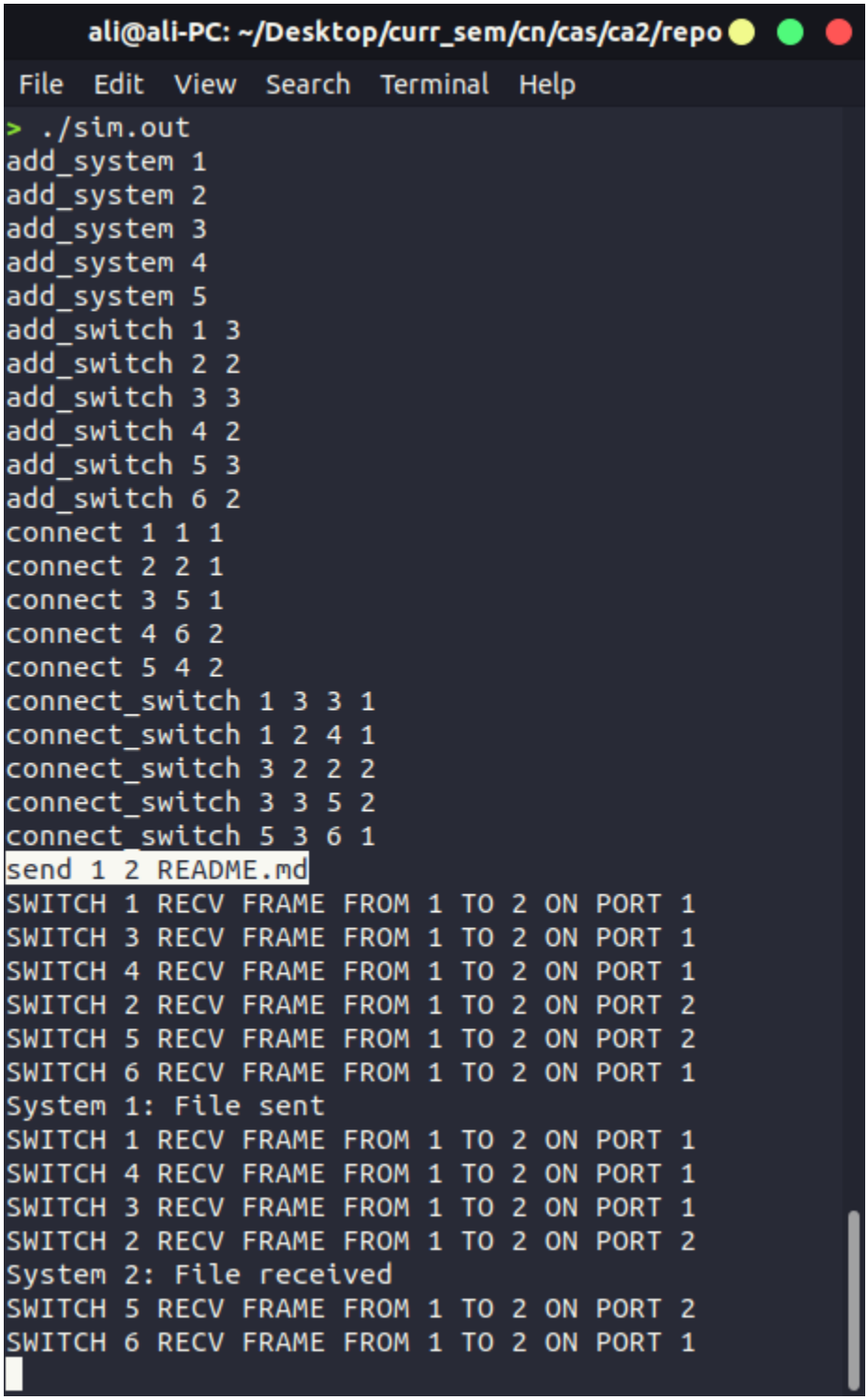
توپولوژی گام ۴



گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

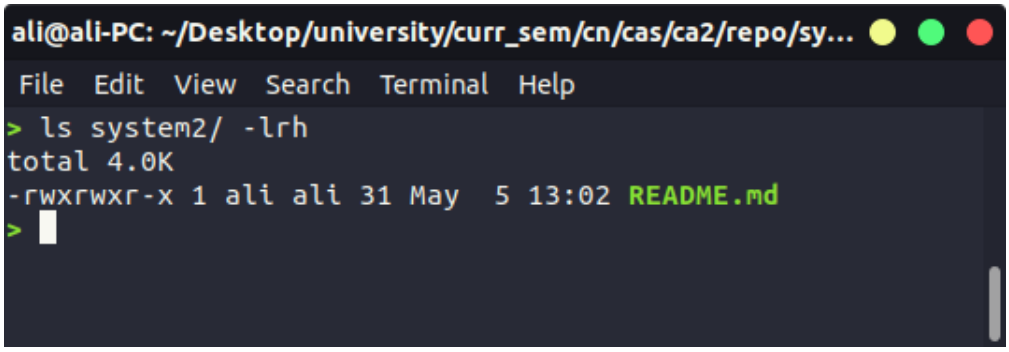
محمدعلی زارع 810197626



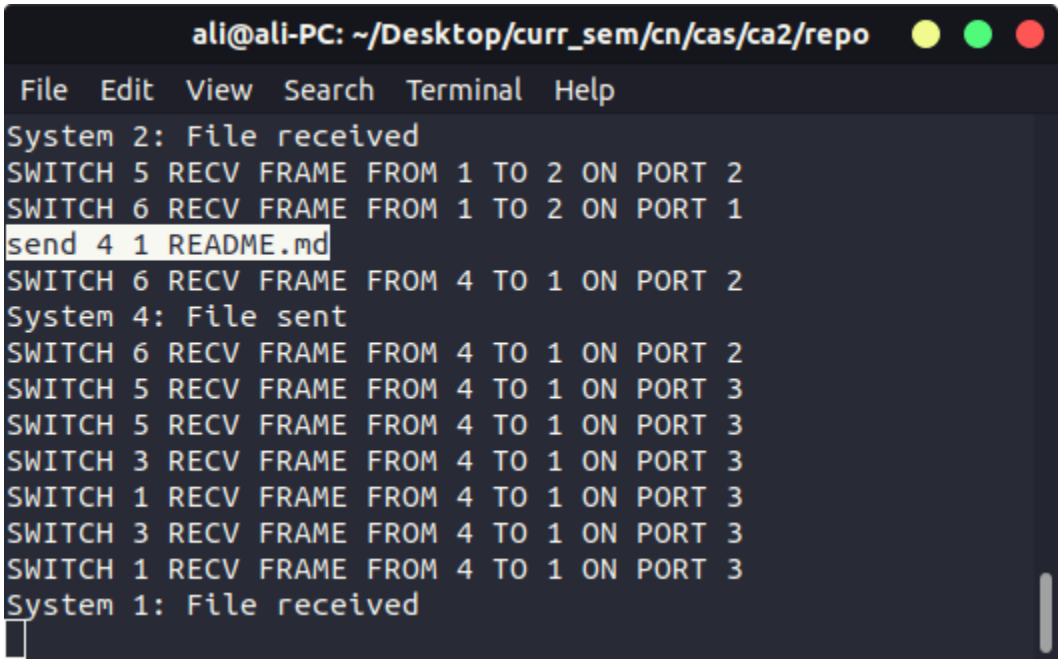
گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626



فریم اول مشخصات فایل و فریم دوم خود فایل است. همینطور که در خروجی دیده می‌شود به دلیل خالی بودن لوک‌آپ تیبل سوییچ‌ها، فریم‌ها در همه سوییچ‌ها پخش شده‌اند. ولی در ادامه شبیه‌سازی، همه سوییچ‌ها مسیر ارسال به سیستم ۱ را می‌دانند و دیگر روی همه پورت‌های خود فریم‌ها را نمی‌فرستند:

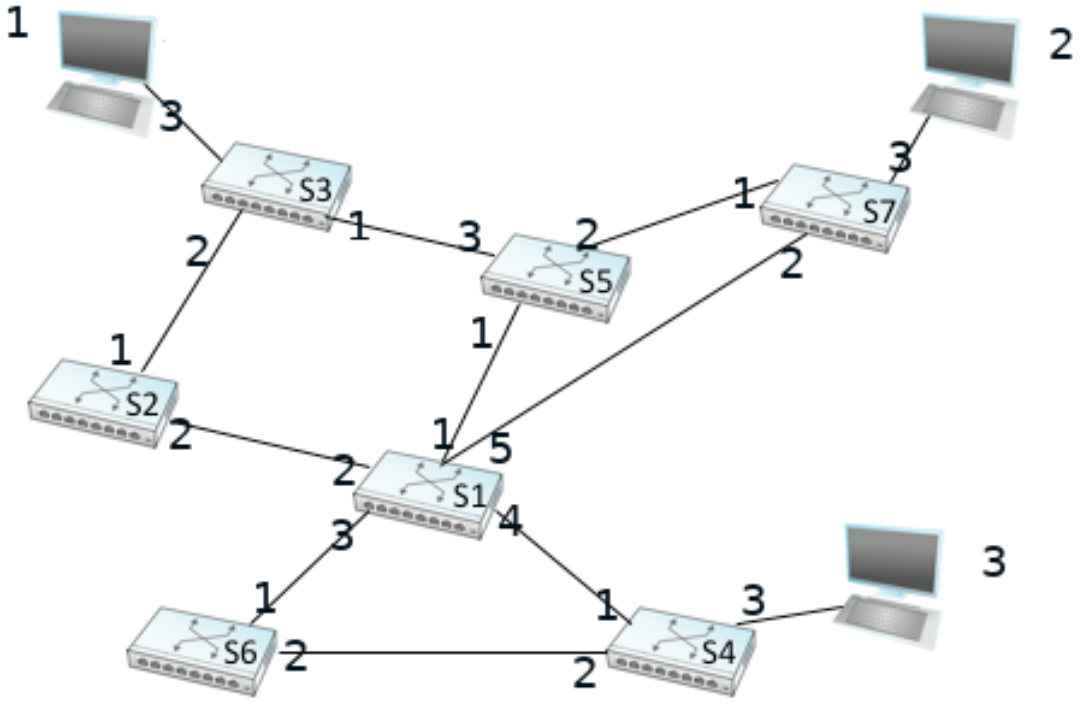


گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

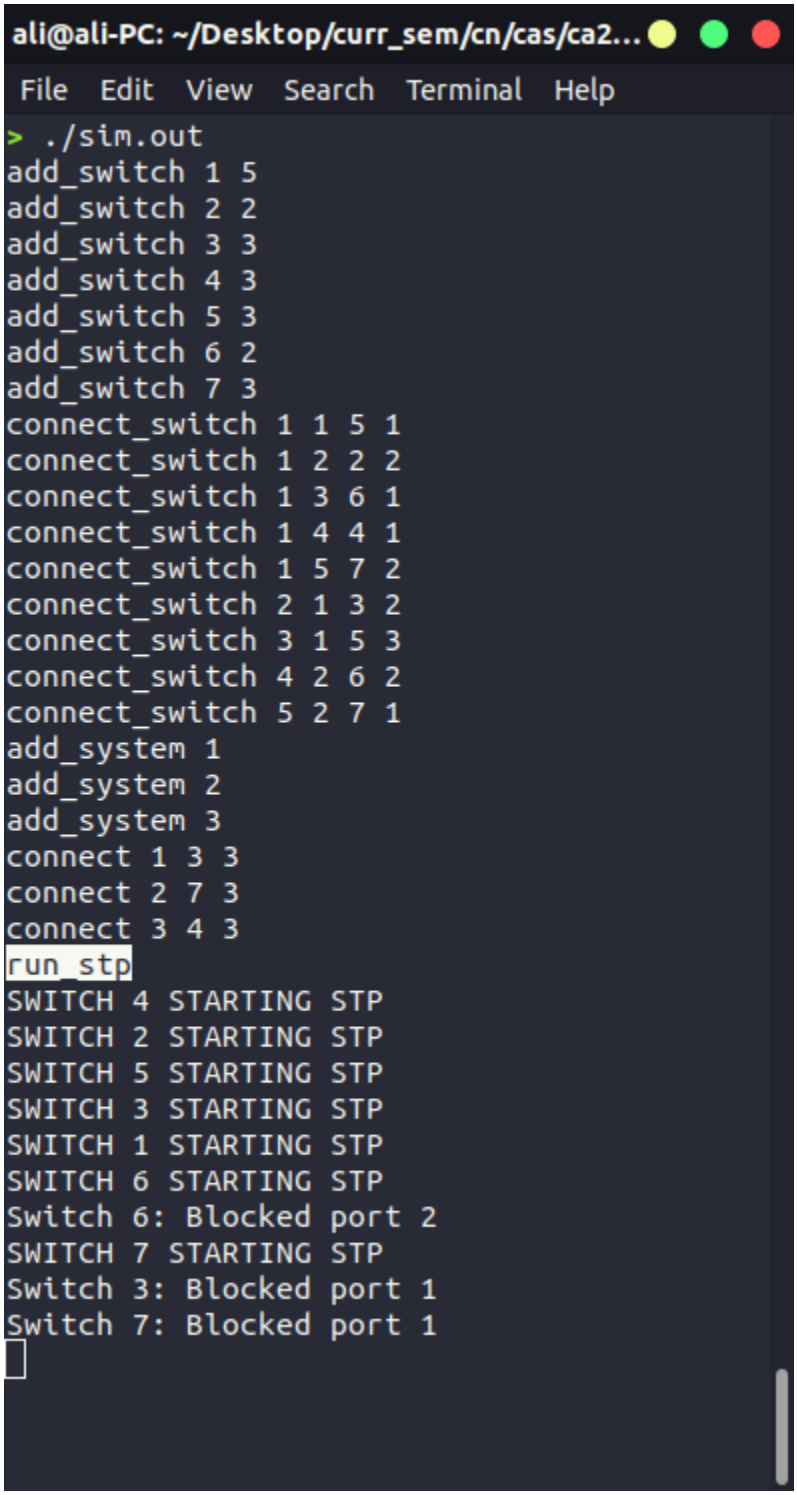
توپولوژی گام ۵ و STP (مثال کتاب)



گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

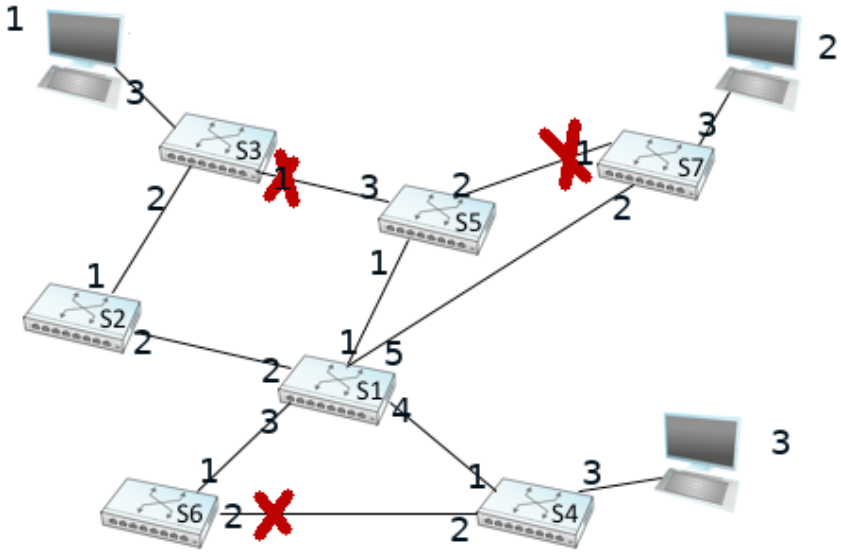


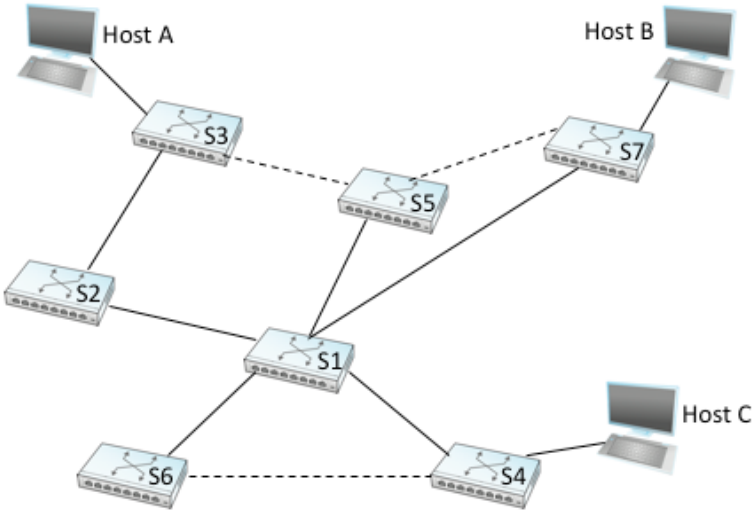
گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

می‌بینیم که سه پورت بلاک شده‌اند. توپولوژی بعد از اجرای الگوریتم:





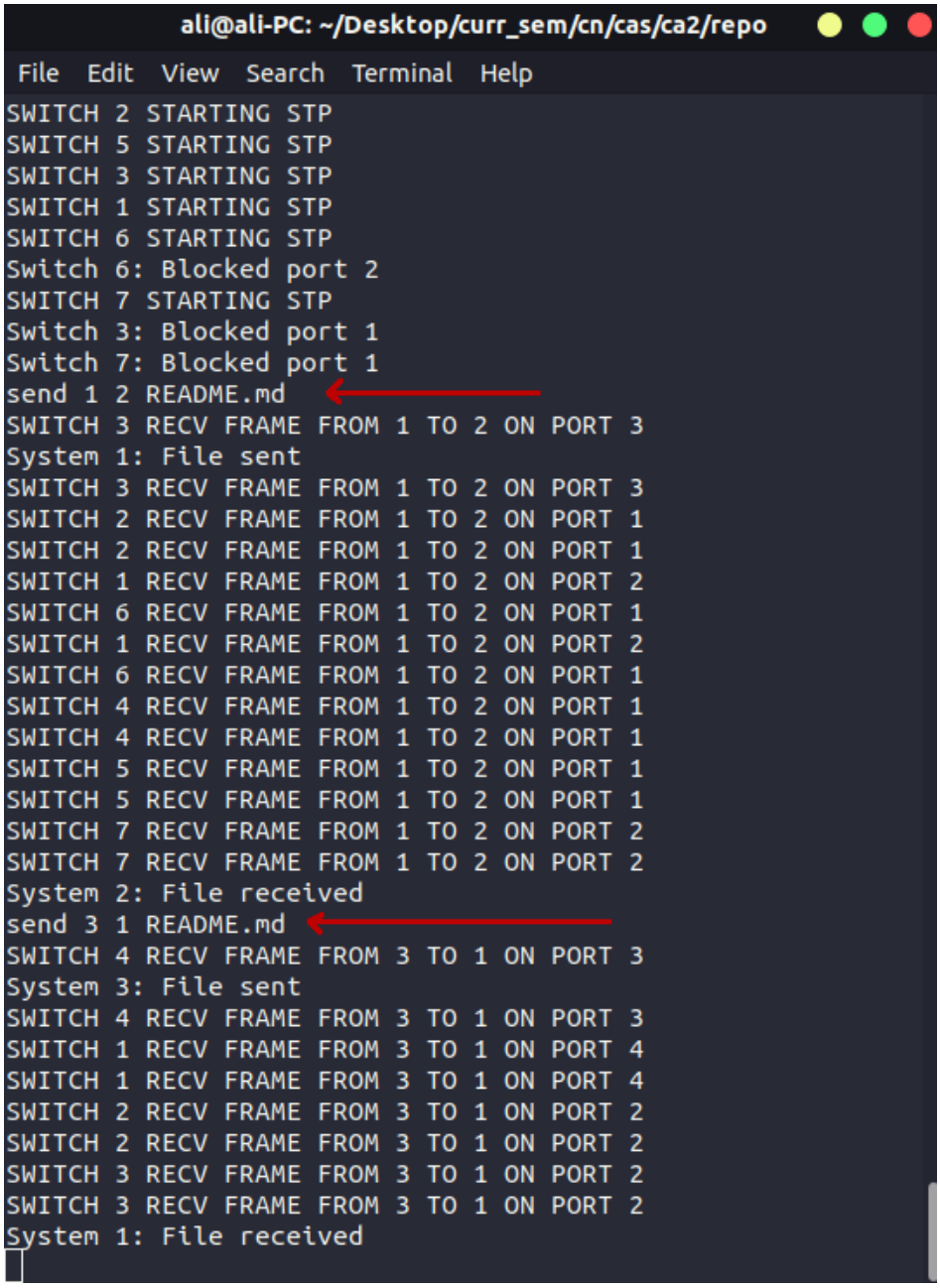
حالا دو بار فایل ارسال می‌کنیم:

باز هم بار اول لوک‌آپ تیبل‌ها خالی هستند.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626



گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

کد

Interface

با اجرا شدن برنامه Interface، در تابع main که درون فایل interface.cpp قرار دارد یک instance از کلاس Interface ساخته می شود و متد run این کلاس صدا زده می شود. در این کلاس دو map برای نگه داشتن آی‌دی سیستم‌ها و سوییچ‌ها و فایل‌دیسکریپتور پایپ دستور مربوط به آن‌ها وجود دارد.

متد run

در این متد در ابتدا دیرکتوری‌ای برای نگهداری named pipe هایی که قرار است ساخته شود ایجاد می شود. سپس در ادامه در یک حلقه بی نهایت از کاربر دستور ورودی می گیرد و آن را اجرا می کند.

متد tokenizeInput:

این متد ورودی استرینگی که می گیرد را به صورت وکتوری از کلمات می شکند و بر می گرداند.

متد addSwitch

در این متد id و تعداد پورت های switch را از آرگومان‌ها می گیرد. و چون interface با switch ها از طریق unnamed pipe ارتباط برقرار می کند بدین صورت سر خواندن این پایپ را به جای ورودی استاندارد این switch قرار می‌دهیم و پردازه سوییچ‌را با آرگومان‌های داده شده می‌سازیم.

همچنین سر خواندن این پایپ را به آیدی سوییچ، map می‌کنیم.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

متد addSystem

در این متد id سیستم را از آرگومان می گیرد. و چون interface با system ها از طریق unnamed pipe ارتباط برقرار می کند بدین صورت سر خواندن این پایپ را به جای ورودی استاندارد این system قرار می‌دهیم و پردازه سیستم‌را با آرگومان‌های داده شده می‌سازیم.

همچنین سر خواندن این پایپ را به آیدی سیستم، map می کنیم.

متد connect

در این متد id سیستم و id سوییچ و شماره پورتی که باید وصل شوند را از آرگومان ها می گیرد سپس نام named pipe های مربوط به سوییچ که سیستم باید به آن متصل شود را برای سیستم می‌فرستد.

متد connectSwitch

برای اتصال دو سوییچ است که نام پایپ هرکدام از سوییچ‌ها و پورت مربوطه‌را برای دیگری می‌فرستد تا جای پایپ out آن پورت خود قرار دهند.

متد sendFile

در این متد مبدا و مقصد و نام فایل را از آرگومان ها می گیرد. سپس یک command قراردادی برای سیستم مربوطه می فرستد.

متد recvFile

در این متد مبدا و مقصد و نام فایل را از آرگومان ها می گیرد. سپس یک command قراردادی برای سیستم مربوطه می فرست.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

متد runStp

این متد یک command قراردادی برای همه ی سوییچ ها می فرستد تا آن ها الگوریتم stp را اجرا کنند.

System

با اجرا شدن برنامه System، در تابع main که درون فایل System.cpp قرار دارد یک instance از این کلاس ساخته می شود و متد run این کلاس صدا زده می شود. در آرگومان main آیدی سیستم مربوطه گرفته می شود. در کانستراکتور این کلاس یک FD_SET برای نگهداری فایل دیسکریپتورهای select ساخته می شود و فایل دیسکریپتور 0 که همون پایپ دستور از سمت interface است قرار داده می‌شود.

در این کلاس آیدی سیستم، و pipe مربوط به خواندن (که این پایپ همان pipe نوشتن سوئیچ متصل است.) و یک pipe برای نوشتن (که این پایپ همان pipe خواندن سوئیچ متصل است.) نگهداری می شود.

متد run

در این متد در حلقه بی نهایت منتظر دریافت پیامی از دو تا فایل دیسکریپتور موجود خواهد ماند. به وسیله select می فهمد که کدام فایل دیسکریپتور پیامی به این سیستم فرستاده است.

اگر فایل دیسکریپتور stdIn یا همان پایپ دستور باشد، متد handleStdIn صدا زده می شود و اگر فایل دیسکریپتور پایپ خواندن از سوییچ باشد، متد handleFrame صدا زده می شود.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

متد handleStdIn

در این متد اگر دستور دریافتی برابر با connect باشد یعنی قرار است این سیستم به سوئیچی متصل شود. در نتیجه نام پایپ های خواندن و نوشتن در سوئیچ را می سازیم و آنها را open می کنیم و فایل دیسکریپتور خواندن از پایپ را در FD_SET اضافه می کنیم. اگر دستور دریافتی send بود متد sendFile را با آرگومان های آیدی گیرنده و اسم فایل صدا می زنیم و اگر دستور دریافتی recv بود متد recvFile را با آرگومان های آیدی فرستنده و اسم فایل صدا می زنیم.

متد isFileReq

این متد چک می کند که اگر عدد فرستاده شده به عنوان سایز فایل برابر با عدد قراردادی 9999 بود یعنی قرار است برای این سیستم فایلی فرستاده شود.

(در فایل Constants.cpp که در فولدر others قرار دارد، این قرارداد تعریف شده است.)

متد handleFrame

به وسیله متد readNumber (که در فایل Utils.cpp در فولدر others تعریف شده است.) از header مقصد و مبدا را می‌خوانیم.

اگر مقصد فریم برابر آیدی این سیستم نبود هیچ کاری انجام نمی‌دهد. (یعنی lookup table سوئیچ هنوز کامل نشده است و برای همه پورت‌های خود فرستاده است)

سپس در ادامه به شرطی که آیدی مقصد و آیدی سیستم مربوطه برابر بود، دو حالت رخ خواهد داد، یا درخواست دریافت فایل از این سیستم فرستاده شده است و یا قرار است فایلی برای این سیستم فرستاده شود.

به وسیله متد isFileReq بررسی می شود که اگر درخواست فایل از این سیستم داده شده بود به آیدی آن سیستم فایل فرستاده شود. در غیر این صورت یعنی قرار است فایلی برای این سیستم فرستاده شود که متد recvFile صدا زده می شود.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

متد sendFile

آرگومان های این متد، آیدی مقصد و نام فایل هستند.

در ابتدا فایلی که نامش را دریافت کرده، باز می کند و سپس سایز آن را محاسبه می کند.

در ادامه در فریم و طبق قرارداد ذکر شده برای فریم، آیدی مقصد و آیدی مبدا را در هدر می‌نویسد. سپس نام فایل و سایز فایل را در فریم می‌نویسد. و سپس سایز پیام را در هدر می‌نویسد. حال مشخصات فایل را می‌فرستد.

در ادامه تا وقتی که دیتای فایل به پایان نرسیده است، دیتا را به اندازه فریم می‌شکند و با هدر مناسب ارسال می‌کند.

متد recvFile

آرگومان های این متد نام فایل و سایز فایل هستند.

در این متد در ابتدا، فولدری به آیدی این سیستم و در فولدر سیستم ایجاد می شود. و سپس فایلی به اسم فایلی که قرار است دریافت شود باز می شود و اگر از قبل موجود نبود، ابتدا ساخته و سپس باز می شود.

در ادامه به اندازه فایل دریافتی و 1024 تا 1024 تا یعنی به اندازه فریم، دیتا دریافت کرده و در فایلی که باز کرده ایم محتوای payload را می‌نویسیم. در پایان کار فایل را می بندیم.

متد sendFileReq

آرگومان های این متد، آیدی مقصد و نام فایل هستند. این متد از این سیستم درخواستی را برای دریافت فایل از سیستم مقصد، می فرستد.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

در این متد و در فریمی و طبق قرارداد ذکر شده برای فریم، آیدی مقصد و آیدی مبدا و عبارت قراردادی 9999 را به عنوان سایز در فریم می‌نویسد. در انتها اسم فایل درخواستی را می‌نویسد و فریم را ارسال می‌کند.

Switch

با اجرا شدن برنامه Switch، در تابع main که درون فایل Switch.cpp قرار دارد یک instance از این کلاس ساخته می شود و متد run این کلاس صدا زده می شود. در آرگومان main آیدی سوئیچ و تعداد پورت‌ها گرفته می شود.

در این کلاس آیدی سوییچ و تعداد پورت ها و یک map برای lookup table از آیدی سیستم به پورت مربوطه و یک map برای writePorts از از شماره پورت به write port یک map برای readPorts از read port به شماره پورت نگهداری می شود.

همچنین یک استراکت STPconfig برای نگه‌داری اطلاعات مربوط به STP و یک set برای نگهداری پورت های بلاک شده خواهیم داشت.

در کانستراکتور این کلاس یک FD_SET برای نگهداری فایل دیسکریپتورهای select ساخته می‌شود و اولین فایل دیسکریپتور که برای خواندن دستور از interface است قرار داده می شود. سپس به ازای هر پورت، دو تا named pipe و یکی برای خواندن و یکی برای نوشتن ساخته می شود که در فولدر fifos قرار داده می شوند و باز می شوند. سپس فایل دیسکریپتور آن‌ها در map های مربوط به خودذخیره می‌شوند.

در آخر فایل دیسکریپتور مربوط به خواندن این سوییچ در FD_SET مربوط به سوییچ قرار داده می شود.

متد run

در این متد در حلقه بی‌نهایت منتظر دریافت پیامی از دو تا فایل دیسکریپتور موجود خواهد ماند. به وسیله select می فهمد که کدام فایل دیسکریپتور پیامی به این سیستم فرستاده است.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

اگر فایل دیسکریپتور stdIn یا همان پایپ دستور باشد، متد handleStdIn صدا زده می شود و اگر فایل دیسکریپتور پایپ پورت های سوئیچ پیامی فرستاده باشد، متد handlleFrame صدا زده می شود.

متد handleStdIn

در این متد اگر دستور دریافتی برابر با connects باشد یعنی قرار است این سوییچ را به سوئیچی دیگر متصل شود. پس در ابتدا پایپ های مربوط به نوشتن سوییچ ها بسته می شود و سپس هر سوییچ، پایپ مربوط به خواندن سوییچ دیگر را باز می کند و به جای پایپ مربوط به نوشتن خودش قرار می دهد و در map مربوط به writePorts آن را به روز رسانی می کند.

و اگر دستور دریافتی برابر با stp بود، آنگاه متد broadcastStp صدا زده می شود.

متد handleFrame

اگر فریم مربوط به پیام‌های STP باشد متد مربوط به آن صدا می‌شود و در غیر این صورت ابتدا لوک‌آپ تیبل آپدیت می‌شود و سپس فوارد می‌کند.

متد updateLookupTable

آرگومان های این متد، آیدی سوییچ و پورت مربوط به این سوییچ است.

در ابتدا بررسی می کند که در map مربوط به lookupTable، آیدی این مبدا وجود دارد یا خیر که اگر وجود نداشت، لوک‌آپ تیبل خودرا آپدیت می‌کند.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

متد forwardFrame

آرگومان های این متد، مبدا و مقصد و فریم است.

این متد بررسی می کند که اگر آیدی مقصد در lookupTable این سوییچ بود، آنگاه فریم را فقط برای پورت مقصد فوروارد می‌کند و در غیر این صورت برای همه ی پورت‌ها جز پورت مبدا، فوروارد می‌کند.

متد handleStpMessage

مشخصات پیام را می‌خوانیم و با متد isItBetter بررسی می‌کنیم که این پیام اطلاعات بهتری مربوط به روت دارد یا خیر. اگر بهتر بود سوییچ اطلاعات STP خود را آپدیت می‌کند و اطلاعات جدید را برای بقیه پورت‌های خود می‌فرستد. در غیر این صورت با متد isItDisignated بررسی می‌کنیم که کدام سوییچ در این اتصال designated است. اگر سوییچ دیگر designated بود، آن پورت را بلاک می‌کند. سپس در آخر متد broadcastStp صدا زده می شود.

متد broadcastStp

سوییچ در این متد پیام قراردادی stp را برای همه ی پورت هایش در صورتی که پورت بلاک شده نباشد و پورت در مسیر root فعلی نباشد، می فرستد.

استراکت STPconfig

در این استراکت، مقدار آیدی root و مقدار آیدی فرستنده و مقدار هزینه تا root و پورتی که از آن به root مسیر داریم را نگه می دارد. این استراکت چند متد هم دارد.

گزارش پروژه دوم درس شبکه های کامپیوتری

حمیدرضا خدادادی 810197499

محمدعلی زارع 810197626

متد isItBetter

این متد بین دو مسیری که از این سوییچ و آیدی سوییچ دریافتی با اولویت های زیر، مسیر بهتر را پیدا می‌کند.

اگر آیدی root هر دو برابر و هزینه رسیدن به root هر دو برابر بود، آنگاه بر اساس آیدی فرستنده مسیر بهتر را پیدا می‌کند. و اگر فقط آیدی root هر دو برابر بود، آنگاه بر اساس آیدی سوییچ ها مسیر پیدا می‌کند. و در غیر این دو حالت بر اساس آیدی root این دو سوییچ مسیر بهتر را پیدا می‌کند.

متد set

این متد اطلاعات استراکت STPconfig را به روز می کند.

متد isItdesignated

این متد بررسی می‌کند که با توجه به پیام دریافتی، سوییچ ارسال کننده در این اتصال designated است یا خیر.

متد makeStpFrame

این متد یک فریم قراردادی برای stp message با توجه به اطلاعاتش می‌سازد.

Utils

تعدادی تابع برای کمک به کارهایی مثل نوشتن عدد در مکانی از فریم یا نوشتن اسم و فایل و سایز و ... دارد.