دلال پیام - قسمت دوم

پرهام الوانی

۴ دی ۱۴۰۰

فهرست مطالب

۱ مقدمه

در قسمت اول پروژه با دلال پیام آشنا شدید. در این قسمت قصد داریم این سامانه را در شرایط مختلف و به منظور یادگیری مفاهیم کنترل جریان و ازدحام بررسی کنیم.

۲ کنترل جریان با TCP

سیستم دلال پیام را همانطور که از قسمت پیشین توسعه داده‌اید، اجرا کنید، یک کلاینت بنوسید که روی یک تاپیک خاص مشترک شده و ارتباط TCP خود را باز نگهدارد، تا به اینجا این کلاینت دقیقا مشابه با کلاینتی است که پیشتر برای اشتراک روی یک تاپیک توسعه داده بودید، اما این کلاینت هیچ داده‌ای را از روی ارتباط TCP **نمی‌خواند**. کلاینت دیگری اجرا کنید که روی این تاپیک خاص شروع به انتشار اطلاعات کند، این انتشار می‌بایست به صورت تکراری ادامه پیدا کند. در این شرایط این ارتباط را در نرم‌افزار Wireshark بررسی کنید، بعد از گذشت چه مدت زمان داده‌ای منتقل نمی‌شود؟ آیا دلال پیام شما در این شرایط خطا می‌دهد؟

۳ بیشترین تعداد ارتباط همزمان TCP

دلال پیام را همانطور که از قسمت پیشین توسعه داده‌اید، اجرا کنید. همانطور که پیش اشاره شد، کلاینت‌هایی که مشترک می‌شدند یک ارتباط باز با این دلال پیام را نگهداری می‌کردند. تعداد این کلاینتها را افزایش دهید، در نظر داشته باشید که نیازی به کلاینت برای انتشار پیام نیست و تنها افزایش تعداد کلاینت‌های متشرک شونده کفایت می‌کند. تا چه میزان کلاینت می‌توانید داشته باشید؟ آیا این تعداد با حداکثر تعداد پورت‌های موجود در ارتباط TCP قابل مقایسه است؟ چه عاملی باعث این محدودیت می‌شود؟

۴ پیاده‌سازی دلال پیام با UDP

قصد داریم دلال پیامی که پیشتر نوشته‌ایم، در کنار پشتیبانی از ارتباط TCP، از ارتباط UDP هم پشتیبانی کند. تفاوت اصلی در ارتباط UDP نبود یک ارتباط پایدار میان سرور و کلاینت است و این مساله در زمان مشترک شدن کلاینتها روی تاپیک‌ها بیشتر خود را نشان می‌دهد چرا که سرور می‌بایست پیام‌ها را برای مشترکین فعال ارسال کند. در ادامه به این موضوعات پرداخته می‌شود.

۱.۴ پیاده‌سازی دلال پیام با استفاده از UDP

مانند قبل سرور می‌بایست بر روی یک پورت و آدرس مشخص قابل دسترسی باشد از این رو سوکتی که برای سرور در نظر گرفته می‌شود با دستور Bind روی یک آدرس و پورت مشخص بسته می‌شود.

دقت داشته باشید که کار با پروتکل UDP بسیار ساده‌تر از TCP است چرا که شما اینجا با بسته‌ها سر و کار دارید و مثلا برای پشتیبانی از چند ارتباط همزمان نیاز به استفاده از Threadها ندارید می‌توانید بسته‌های ارتباط‌های مختلف را با استفاده از یک فراخوانی تابع دریافت کنید.

۲.۴ اطمینان از سلامت طرف مقابل

در این حالت پروتکل مشابه آنچه پیشتر آورده شد باقی می‌ماند اما برای پیدا کردن مشترکین فعال نیاز داریم تا از پیام‌های Ping و Pong استفاده کنیم. این پیام‌ها به سرور و کلاینت اجازه می‌دهند بر پایه ارتباط UDP از فعال بودن طرف مقابل اطمینان حاصل کنند.

بنابراین پیاده‌سازی پیام‌های Ping و Pong در این پیاده‌سازی پروتکل UDP اجباری هستند و شما از این طریق می‌توانید کلاینتهایی که ارتباطشان قطع شده است را تشخیص دهید. برای این امر هر طرف در دوره‌های ۱۰ ثانیه‌ای پیام‌های Ping را ارسال می‌کند و اگر تا شروع دوره‌ی بعدی پاسخی دریافت نکند به منزله خطا در اتصال خواهد بود.

۳.۴ ذخیره کردن اطلاعات طرف مقابل

کلاینت‌های UDP برای ارتباط با سرور تنها نیاز به پورت و آدرس دلال پیام دارند، اما سرور برای ارسال پیام به کلاینت‌ها می‌بایست از آخرین پورت و آی پی استفاده کند که با آن از کلاینت پیام دریافت کرده است. در نظر داشته باشید که برای این امر نیاز به دستور خاصی ندارید چرا که به صورت پیشفرض یک پورت تصادفی برای کلاینت شما انتخاب شده و می‌تواند از آن برای ارسال یا دریافت اطلاعات استفاده کند.

```python
import socket

client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # UDP
while True:
    data, addr = client.recvfrom(1024)
    print("received message:",data,addr)
```

در کد نمونه آورده شده سوکت UDP به یک پورت آزاد و تصادفی مانند پورت ۳۷۰۲۰ بسته خواهد شد و به این ترتیب بسته‌های ارسال از پورت مبدا ۳۷۰۲۰ ارسال شده و از سوی دیگر می‌توان روی این پورت مطابق آنچه در مثال آورده شده است بسته دریافت نمود.

۵ مقایسه کنترل جریان در TCP و UDP

در قسمت قبلی با استفاده از کلاینت TCP که از ارتباط خود چیزی دریافت نمی‌کرد بحث کنترل جریان در پروتکل TCP را دیدید. در این قسمت قصد داریم همین موضوع را در پیاده‌سازی UDP ببینیم. برای این موضوع، کلاینت UDPای دارید که روی یک موضوع مشترک شده است ولی بسته‌ای را **نمی‌خواند** دقت داشته باشید که یعنی پس از ارسال و دریافت بسته‌های لازم جهت اشتراک بر یک موضوع خاص، دیگر بسته‌ای را دریافت نمی‌کند. یک کلاینت دیگر به صورت تکراری شروع به انتشار در همان موضوع می‌کند. با استفاده از نرم‌افزار Wireshark توضیح دهید چه اتفاقی برای بسته‌های می‌افتد. آیا این بسته‌ها از دست می‌روند؟