

دلال پیام - قسمت دوم

پرهام الوانی

۴ دی ۱۴۰۰

فهرست مطالب

۲	۱ مقدمه
۲	۲ کنترل جریان با TCP
۲	۳ بیشترین تعداد ارتباط همزمان TCP
۲	۴ پیاده‌سازی دلال پیام با UDP
۲	۱.۴ اطمینان از سلامت طرف مقابل
۲	۲.۴ ذخیره کردن اطلاعات طرف مقابل

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 بنویسید که روی یک تاپیک خاص مشترک شده و ارتباط TCP خود را باز نگهدارد، تا به اینجا این کلاینت دقیقاً 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 مشترک 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.

۱.۴ 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.

۲.۴ 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. پس شما می‌بایست در هنگام توسعه کلاینت با UDP به این نکته توجه داشته باشید ابتدا روی یک پورت مشخص Bind کنید تا سرور بتواند شما را پیدا کند.

```
import socket

client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # UDP
client.bind(("", 37020))
while True:
    data, addr = client.recvfrom(1024)
    print("received message:", data, addr)
```

در کد نمونه آورده شده سوکت UDP به پورت ۳۷۰۲۰ بسته شده است و به این ترتیب بسته‌های ارسال از پورت مبدأ ۳۷۰۲۰ ارسال شده و از سوی دیگر می‌توان روی این پورت مطابق آنچه در مثال آورده شده است بسته دریافت نمود.

این سند برپایه بسته X3Persian گونه 23.1 توسعه پیدا کرده است.