به نام خدا

تمرین ششم سیستم عامل

زهرا رحیمی ۹۸۳۱۰۲۶

۱.

الف) MulticastSocket خود زیر کلاسی از DatagramSocket است و در ساختار آن از DatagramSocket استفاده شده است

ب) از آنجا که همه پورت های کوچکتر از ۱۰۲۴ شناخته شده هستند و از آنها برای پیاده سازی سرویس های مختلف استفاده می کنیم، پورت باید بزرگتر از ۱۰۲۴ باشد و چون همه ارتباطات باید منحصر به فرد باشند و در سوال گفته شده که فرآیند جدید از پورت ۱۶۰۰ با همان وب سرور می خواهد ارتباط برقرار کند، پورت نباید برابر ۱۶۰۰ باشد.

ج) زیرا سوکت اجازه انتقال اطلاعات بین چندین رشته(thread) را به صورت غیر ساختارمند می دهد و وظیفه کلاینت یا سرور است که به آن دیتا و اطلاعات را ساختار و فرم دهند. در حالیکه در متد های سطح بالاتر remote procedure calls(RPCs) و pipes این مشکل برطرف شده است.

۲.

الف) ordinary pipe

ب) named pipe: ارتباط در named pipe برخلاف ordinary که یک طرفه است، می تواند دوطرفه می باشد. رابطه پدر-فرزندی بین فرآیندها نیاز نیست و ارتباط می تواند بین چندین فرآیند باشد(شبکه ای از فرآیندها). و نکته دیگر هم این که pipe بعد از اتمام ارتباط فرآیند ها باقی می ماند و از بین نمی رود

ج) named pipe

د) ordinary pipe

۳.

Speedup = 1 / (S + (1-S)/N)

از آنجا که ۸۰ درصد تسریع پیدا کرده پس سرعت آن ۱.۸ برابر شده است و S = 0.1 است زیرا ریسمان ها برای آن ۹۰ درصد از برنامه که به صورت موازی اجرا می شوند نیاز است:

$$1.8 <= 1 / (0.1 + 0.9/N); \quad N >= 1.97 \sim 2$$

در نتیجه حداقل ۲ ریسمان باید استفاده کنیم.

۴.

الف) مدیریت thread ها در مدل many to one بهینه است زیرا این کار در فضای کاربر با استفاده از thread library انجام می گیرد.

در مدل many to many محدودیتی برای تعداد ریسمان های هسته وجود ندارد و حتی در مدل دو سطحی از آن می توان یک ریسمان از کاربر را به یک ریسمان از هسته اختصاص داد اما در مدلی از آن باید تعداد ریسمان های هسته کمتر از ریسمان های کاربر باشد. این مدل با اینکه هم مشکل سربار در مدل one to one را ندارد و هم از مزیت هم روندی اجرای رشته ها در مدل many to one بهره می برد کمتر در سیستم عامل های امروزی استفاده می شود زیرا علاوه بر اینکه پیاده سازی آن دشوار است، با ارزان تر شدن حافظه هسته و افزایش تعداد هسته ها در پردازنده ها محدودیت تعداد ریسمان کمتر وجود دارد.

در مدل many to one از آنجا که همه ریسمان های user تنها با یک ریسمان از هسته می توانند تعامل داشته باشند، منابع کمتری استفاده می شود ولی این عیب را دارد که thread ها نمی توانند به صورت هم روند از سیستم عامل اجرا شوند و باید منتظر یکدیگر بمانند تا کار آن یکی تمام شود تا این شروع به کار کند و block شدن یک رشته باعث block شدن همه رشته ها می شود به همین دلیل این مدل در سیستم عامل های خیلی کمی مورد استفاده قرار می گیرد

ب) از آنجا که در مدل one to one به ازای هر ریسمان در user thread یک ریسمان در kernel thread تولید می شود، استفاده زیاد از منابع هسته و سربار زیاد می شود در نتیجه توسعه دهنده باید در مورد تعداد ریسمان های user احتیاط کند تا سربار به وجود نیاید. علاوه بر آن به دلیل گفته شده در پیاده سازی این مدل محدودیت هایی برای تعداد user thread هایی که سیستم عامل می تواند ساپورت کند، گذاشته شده است.

ج) در مدل many to many مدیریت ریسمان ها بهتر است زیرا هم مشکل سربار در مدل one to one را ندارد و هم از مزیت هم روندی اجرای رشته ها در مدل many to one بهره می برد. اما ارزان تر شدن حافظه هسته و افزایش تعداد هسته ها در پردازنده ها و استفاده همه گیر از pthread باعث کم تر استفاده شدن از این مدل در سیستم های عامل امروزی شده است