به نام خدا

آزمایشگاه سیستم عامل

تمرین کامپیوتری سوم: همگام سازی

تاریخ تحویل: ۲۰ آذر

اهداف پروژه:

- آشنایی با لیست پیوندی هسته لینوکس[1]
- آشنایی با ساختار داده task_struct
- آشنایی با سمافور[2]موجود در هسته لینوکس
- آشنایی با نمونه‌ای از پیاده سازی مکانیسم های همگام سازی
- پیاده سازی و اشکال‌زدایی یک مکانیسم جدید در هسته‌ی لینوکس

چکیده:

در این پروژه شما با جزییات پیاده سازی یکی از مفاهیمی که در کلاس یادگرفته بیشتر آشنا میشوید. در ادامه با استفاده از دانش بدست آمده از قسمت اول و پروژه های قبلی، خودتان یک نمونه جدید پیاده سازی میکنید و آن را در برنامه‌ای در سطح کابر مورد استفاده قرار میدهید.

قسمت اول: آشنایی با سمافور هسته لینوکس

در این قسمت ابتدا به بررسی کد منبع مکانیزم همگام سازی[3]هسته لینوکس بپردازید. برای این کار میتوانید تعاریف و پیاده سازی های مرتبط با سمافور را از فایل های semaphore.h[4] و semaphore.c[5] بیابید و در ادامه سوالات زیر را در گزارش خود پاسخ دهید.

۱. توضیح دهید هر عضو ساختار داده اصلی سمافور در فایل semaphore.h چه کاربردی دارد.

---

[1] struct list_head
[2] semaphore
[3] synchronization
[4] include/linux/semaphore.h
[5] kernel/locking/semaphore.c

۲. با توجه به نحوه ی پیاده سازی توابع ()up و ()down در فایل semaphore.c توضیح دهید این دو تابع چه راه حلی برای محافظت از ناحیه بحرانی اتخاذ میکنند.

۳. نحوه ی عملکرد تابع ()up در ناحیه بحرانی را شرح دهید. (توضیح مختصری در رابطه با توابع داخلی نیز بدهید.)

۴. نحوه ی عملکرد تابع()down در ناحیه بحرانی را شرح دهید. (توضیح مختصری در رابطه با توابع داخلی نیز بدهید.)

قسمت دوم: پیاده سازی یک سمافور جدید

حال میخواهیم یک سمافور جدید با روش افزایش تصادفی[1]پیاده سازی کنیم. برای پیاده سازی سمافور جدید میخواهیم اولویت را در انتخاب پردازه بعدی برای ورود به سمافور دخیل کنیم بدین ترتیب که در صورتی که یک پردازه (که در اینجا میشود پراولویت ترین پردازه پشت سمافور)، منتظر ورود به ناحیه بحرانی بود به صورت رندوم یکی از پردازه های داخل ناحیه بحرانی که اولویت کمتری از خودش دارد را انتخاب میکند و برای مدت زمان معینی اولویت آن را هم اندازه اولویت خودش میکند سپس بعد از گذشت آن مدت زمان دوباره اولویت آن پردازه را به اولویت ابتدایی اش بر میگرداند، حال درصورتی که پراولویت ترین پردازه هنوز وارد ناحیه بحرانی نشده بود همین عمل دوباره تکرار میشود. (این روش کاملا مشابه روش وراثت اولویت[2]اما هر دفعه فقط یک پردازه را برای مدت زمان معینی که فکر میکنیم که در آن مدت زمان ناحیه بحرانی را کامل اجرا میکند، اولویتش را افزایش میدهیم.)

۵. روش های افزایش تصادفی و وراثت اولویت را از نظر وارونگی اولویت،[3] گرسنگی[4]و بن‌بست[5]با هم مقایسه کنید.

پیاده سازی شما باید تایپ جدید mysem را به همراه 3 فراخوان سیستمی[6] زیر در اختیار کاربر قرار دهد:

- int mysem_init(mysem* instance, int n): اشاره‌گر به یک mysem را دریافت کرده و مقداردهی اولیه‌ی آن را انجام میدهد. n نشان‌دهنده تعداد پردازه هایی است که به صورت همزمان میتوانند در ناحیه بحرانی باشند.

[1] Random Boosting
[2] Priority Inheritance
[3] Priority Inversion
[4] Starvation
[5] Deadlock
[6] system call

- int mysem_down(mysem* instance): مقدار سمافور را کاهش میدهد و اگر مقدار منفی شود پردازه‌ی درخواست دهنده را به حالت sleep می‌برد.

- int mysem_up(mysem* instance): مقدار سمافور را افزایش میدهد و در صورت وجود چند پردازه‌ی منتظر پر اولویت‌ترین آن‌ها را انتخاب میکند.

مقدار بازگردانده‌شده توسط این توابع باید نشان‌دهنده‌ی موفقیت فراخوانی و یا کد خطایی که رخداده باشد. (صفر نشان‌دهنده‌ی موفقیت و مقادیر منفی نشان‌دهنده‌ی خطا باشند.)

نمونه استفاده از mysem که باید در برنامه سطح کاربر به شکل زیر است:

```
mysem* ins[1]= new mysem;
int ret_val, n = 3;
ret_val = mysem_init(ins, n);
ret_val = mysem_down(ins);
ret_val = mysem_up(ins);
```

همچنین ***باید در انتخاب ساختمان داده و الگوریتم مورد استفاده ی خود، دقت کنید و آنرا بهینه انتخاب کنید و تحلیل مناسبی برای انتخاب و پیاده سازیتان داشته باشید.***

سایر نکات:

- در کد سطح کرنل و سطح کاربرتان از لاگ های مناسب استفاده کنید.
- طبیعت پروژه‌های آزمایشگاه به گونه‌ای است که ممکن است با مشکلات پیشبینی نشده مواجه شوید، دستیاران آموزشی در رفع این مشکلات به شما کمک خواهندکرد، ولی مسئولیت انجام درست پروژه به عهده ی خود شما است. بنابراین توصیه میشود که پروژه را زود شروع کنید.
- حتما درجلسه‌ی توجیهی حضور داشته باشید. نکاتی که در کلاس یا فروم مطرح میشوند جزء پروژه هستند.
- پروژه‌های آزمایشگاه باید در گروه‌های سه نفره انجام شوند. تمام اعضای گروه باید روی تمام قسمت‌های پروژه تسلط داشته باشند و هریک از اعضا متناسب با میزان تسلط نمره‌دهی خواهد شد.

موفق و سربلند باشید.

[1] **میتواند در برنامه سطح کاربر تایپ ins، *void باشد.**