به نام خدا

آزمایشگاه سیستم‌عامل

تمرین کامپیوتری سوم: همگام‌سازی

تاریخ تحویل: یکشنبه ۱۶ اردیبهشت

اهداف پروژه:

- آشنایی با سمافور[1] و mutex هسته لینوکس
- آشنایی با نمونه‌هایی از پیاده سازی مکانیزم‌های همگام‌سازی
- پیاده‌سازی و اشکال‌زدایی یک مکانیزم جدید در هسته‌ی لینوکس

چکیده:

در این تمرین کامپیوتری برای درک بهتر و عمیق‌تر مفهوم همگام‌سازی ابتدا به بررسی تعدادی از مکانیزم‌های همگام‌سازی کرنل می‌پردازیم، سپس بعد از فهم سازوکارهای به کاررفته در آن سعی میکنیم یک مکانیزم همگام‌سازی ساده و جدید در سطح کرنل ایجاد کنیم. در پایان نیز با سناریوهای مختلف در برنامه‌های سطح کاربر صحت آن را مورد آزمایش قرار می‌دهیم.

بخش اول: بررسی پیاده‌سازی مکانیزم‌های همگام‌سازی در هسته لینوکس

در این قسمت ابتدا به بررسی کد منبع دو نمونه از مکانیزم‌های همگام سازی[2] هسته لینوکس می‌پردازیم. تعاریف و پیاده سازی های مرتبط با سمافور را از فایل های semaphore.h[3] و semaphore.c[4] استخراج کنید و برای پاسخ به سوالات مربوط به mutex می‌توانید از فایل های mutex.h[5] و mutex.c[6] کمک بگیرید.

---

1 semaphore

2 synchronization

3 include/linux/semaphore.h

4 kernel/locking/semaphore.c

5 Include/linux/mutex.h

6 kernel/locking/mutex.c

همچنین توجه داشته باشید که ممکن است برای انجام بررسی های خود فایل های دیگری از کرنل را نیز مورد کنکاش قرار دهید.

❖ در رابطه با سمافور هسته لینوکس به سوالات زیر در گزارش خود پاسخ دهید:

۱. توضیح دهید هر عضو ساختار داده اصلی سمافور در فایل semaphore.h چه کاربردی دارد.
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. (سعی کنید توضیح مختصری از نحوه‌ی نحوه‌ی عملکرد این دو تابع و توابع داخلی و مرتبط با آن‌ها بدهید.)

❖ در رابطه با mutex هسته لینوکس به سوالات زیر در گزارش خود پاسخ دهید:[1]

۳. توضیح مختصری در رابطه با عضو count در ساختار داده اصلی mutex دهید. هر کدام از مقادیر آن به چه معناست؟ همچنین توضیح مختصری در رابطه با نوع داده‌ی این عضو و کاربردها و ویژگی‌های آن بدهید.
۴. هرکدام از ۴ تابع اصلی mutex یعنی mutex_init، mutex_lock، mutex_unlock و mutex_trylock چه کاری انجام می‌دهند؟ توضیح مختصری در رابطه با نحوه عملکردشان دهید.[2]

۵. با بررسی‌هایی که در سوالات قبل انجام دادید، به نظر شما mutex و سمافور کرنل چه تفاوت‌هایی با یکدیگر دارند؟ از هر کدام باید در چه شرایطی استفاده کرد؟

۶. امتیازی: یکی دیگر از مکانیزم های همگام‌سازی موجود در هسته لینوکس spinlock است. نحوه‌ی عملکرد آن را در کرنل توضیح دهید و تفاوت‌های آن را با سمافور و کرنل بیان کنید.

[1] در این بخش نیازی به بررسی تکه کدهای مربوط به حالت دیباگ mutex نیست و درصورت بررسی آن‌ها نمره امتیازی می‌گیرید.
[2] در صورت بررسی توابع درونی این توابع نمره امتیازی می‌گیرید.

بخش دوم: پیاده‌سازی یک مکانیزم همگام‌سازی جدید

حال به پیاده‌سازی یک مکانیزم جدید همگام‌سازی می‌پردازیم و نام آن را mysync می‌گذاریم. mysync با چهار فراخوان سیستمی زیر قابل استفاده است:

- int mysync_make_event():

این فراخوان سیستمی یک event جدید را می‌سازد و شناسه آن را برمی‌گرداند. در صورتی که به هر دلیل این کار را نتوانست انجام دهد ۱- برمی‌گرداند.

- int mysync_destroy_event(int eventID):

شناسه یک event را می‌گیرد و آن event را نامعتبر می‌کند. همه‌ی پردازه‌هایی که منتظر این event بوده باشند از حالت بلاک خارج می‌شوند. در صورتی که عملیات موفقیت آمیز انجام شود، تعداد پردازه‌های که به خاطر این event بلاک شده بودند را برمی‌گرداند و در غیر این صورت ۱- برگشت‌داده می‌شود.

- int mysync_wait_event (int eventID):

شناسه یک event را می‌گیرد. پردازه‌ای که این فراخوان سیستمی را صدا زده باشد بلاک می‌شود. در صورت موفقیت‌آمیز بودن عملیات ۱ و در غیر اینصورت ۱- را برمی‌گرداند.

- int mysync_sig_event(int eventID):

شناسه یک event را می‌گیرد و همه‌ی پردازه‌هایی که منتظر این event بوده باشند از حالت بلاک خارج می‌شوند. در صورتی که عملیات موفقیت آمیز انجام شود، تعداد پردازه‌های که به خاطر این event بلاک شده بودند را برمی‌گرداند و در غیر این صورت ۱- برگشت‌داده می‌شود.

- ✓ توجه داشته باشد که باید در انتخاب ساختمان داده و الگوریتم مورد استفاده‌ی خود دقت کنید و آن‌را بهینه انتخاب کنید و تحلیل مناسبی برای انتخاب و پیاده سازیتان داشته باشید.

بخش سوم: بررسی صحت کد با انجام آزمون‌هایی در برنامه‌های سطح کاربر

حال به آزمودن کد سطح کرنل بوسیله سناریوهای زیر می‌پردازیم:

A. سناریویی که در آن هیچ پردازه‌ای منتظر eventی با شناسه eid نیست و تابع mysync_sig_event (eid) صدا زده می‌شود.
B. سناریویی که در آن پردازه‌ای وجود دارد که منتظر event با شناسه eid هست و تابع mysync_sig_event (eid) صدا زده می‌شود.
C. پردازه‌ای وجود دارد که منتظر event با شناسه eid1 است و پردازه‌ی دیگری وجود دارد که منتظر event با eid2 است. توابع صدا mysync_destroy_event(eid1) و mysync_destroy_event(eid2) صدا زده می‌شوند.

✓ برای نشان دادن صحت آزمون‌های خود سعی کنید از لاگ‌های مناسب در برنامه‌های سطح کاربر خود استفاده کنید.

سایر نکات:

- گزارش شما قسمت قابل‌توجهی از نمره ی شما را تشکیل می‌دهد. گزارش باید شامل جواب سوالات موجود و نحوه‌ی پیاده‌سازیتان باشد.
- برای انجام این آزمایش توصیه می‌شود فصل دهم کتاب Linux Kernel Development را مطالعه کنید.
- در کد سطح کرنل و سطح کاربرتان از لاگ های مناسب استفاده کنید.
- دقت داشته باشید که کد نوشته‌شده در کرنل لینوکس مانند سایر کدها نیاز به خوانایی، تمیزی، ماژولار بودن و ... دارد و عدم رعایت این موارد منجر به کسر نمره میگردد.
- 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.
- 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.
- از طریق ایمیل زیر و فروم درس میتوانید سوالات خود را مطرح کنید:

shayanpakzad@gmail.com

موفق و سربلند باشید.