

پاسخ:

در سیستم‌عامل 6v، فراخوانی‌های سیستمی معمولاً با استفاده از یک دستور trap یا وقفه‌ی نرم‌افزاری به کرنل ارسال می‌شوند. هر کدام از توابع کتابخانه سطح کاربر که به عملکردهای سیستمی نیاز دارند (مانند خواندن و نوشتن فایل‌ها، تخصیص حافظه، یا مدیریت پروسه‌ها)، یک فراخوانی سیستمی مناسب را اجرا کرده و سپس به کرنل منتقل می‌کنند. این توابع، شناسه‌ای منحصر به فرد (شماره‌ی فراخوانی سیستمی) و پارامترهای مورد نیاز را به کرنل ارسال می‌کنند، که کرنل این شناسه را تفسیر و سرویس مورد نظر را اجرا می‌کند. این نوع طراحی، به توابع کتابخانه‌های سطح کاربر اجازه می‌دهد تا بدون نیاز به دسترسی مستقیم به سخت‌افزار و جزئیات داخلی کرنل، به سرویس‌های سیستمی دسترسی پیدا کنند.



Ulib شامل آبجکت فایل های زیر است.به کد سی هر کدام مراجعه می کنیم و درباره آن ها توضیح می دهیم:

Ulib.c:

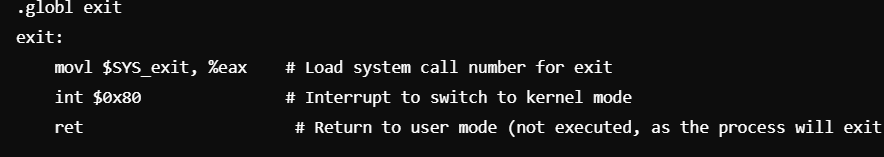
این فایل شامل توابع کاربردی برای کاربر است.این توابع شامل توابع مربوط به تغییر string ها مانند strlrn ، strcpy و strcmp است.گروه دیگر این توابع شامل توابع مربوط به مموری مانند memset و همچنین فانکشن های پرکاربردی برای تبدیل اینت به استرینگ و برعکس مانند atoi است.

در دو تابع gets و stat به ترتیب از syscall های read(برای خواندن از stdin) در اولی،open،close و fstat (اطلاعات فایل ها) استفاده شده.

Usys.s:

شامل یک سری interface برای استفاده از سیستم کال ها در سطح کاربر است.با استفاده از کد سیستم کال و اینتراپت ترنزیشن از سطح یوزر به سطح کرنل اتفاق می افتد.

این تصویر خلاصه ای از کد های در Usys.s است:



Printf.c:

این فایل شامل فانکشن هایی برای نوشتن است.با تابع printint در پروژه ی قبلی برای تغییر کنسول آشنا شدیم.توابعی مانند putc نیز در تابع console برای چاپ کاراکتر استفاده شده است.

سیستم کال استفاده شده در اینجا سیستم کال write است که برای در فایل دیسکریپتور مورد نظر می نویسد.

Umalloc.c:

در اینجا تابع morecore را داریم که از سیستم کال sbrk برای افزایش سگمنت دیتای پراسس برای مقدار خاصی استفاده می کند.این سیستم کال ترمز برنامه(جایی که دیتا سگمنت تمام شده) را تغییر می دهد تا مموری اضافی فراهم شود. تابع mallocنیز از تابع morecore در مواردی استفاده می کند و غیر مسقیم با این سیستم کال سر و کار دارد.

توابعی مانند free نیز وجود دارد که با کرنل کاری ندارد و صرفا حافظه را برای الوکیشن های بعد آزاد می کند.

سیستم کال ها یک اینترفیس فراهم می کنند که مستقل از معماری سخت افزار در صورت ساپورت سیستم عامل از این سیستم کال ها می توان از آن سیستم عامل استفاده کرد.از طرفی برنامه هایی که از سیستم کال ها استفاده می کنند لازم نیست جزییات دقیق مدیریتی سیستم عامل را بدانند و در صورت دانستن اینترفیس سیستم کال ها می توانند به راحتی با هر سیستم عاملی کار کنند.