شروع	پنج‌شنبه، 13 آبان 1400، 5:30 عصر
وضعیت	پایان‌یافته
پایان	پنج‌شنبه، 13 آبان 1400، 5:59 عصر
زمان صرف شده	29 دقیقه 52 ثانیه
نمره	**3.50** از 4.00 (**87.5**٪)

سؤال 1

درست

نمره 1.00 از 1.00

کدام یک از گزینه های زیر صحیح نیست

a. system callها در مد کرنل به صورت اینتراپت اجرا می‌شوند

b. system call ها از طریق برنامه نویسی ماژول کرنل قابل اضافه کردن به کرنل یا حذف از کرنل هستند ✔

c. system call و مد کرنل از اهداف طراحی سیستم عامل برای برآورده کردن protection است

d. تعریف مد کرنل و مد کاربر از طریق سخت افزار امکان پذیر می شود

پاسخ شما صحیح می باشد

پاسخ درست «
system call ها از طریق برنامه نویسی ماژول کرنل قابل اضافه کردن به کرنل یا حذف از کرنل هستند» است.

سؤال 2

درست

نمره 1.00 از 1.00

دیوایس کنترلر بخشی از کرنل است که ارتباط I/O با دیوایس درایور را فراهم می‌کند.

یک گزینه را انتخاب کنید:

- صحیح
- غلط ✔

پاسخ درست گزینهٔ «غلط» است.

سؤال 3

کامل

نمره داده نشده

موازی سازی و همزمانی در سیستمهای کامپیوتری از چه طریق اجرایی میشود؟

a. time sharing رویکردی است که مالتی تسک بودن هر نوع سیستمی را امکان پذیر می‌کند

b. موازی سازی پروسس ها در سیستم مالتی پروسسور از طریق اجرا روی پروسسورهای مختلف و روی سیستم تک پروسسور از طریق time sharing قابل اجرا است

c. موازی سازی و همزمانی پروسس ها در سیستم تک پروسسور از طریق time sharing قابل اجرا است

d. time sharing رویکردی است برای امکان مالتی تسک کردن و موازی سازی پروسسها در سیستم‌های تک پروسسور و چند پروسسور

پاسخ شما صحیح می باشد

پاسخ درست «
time sharing رویکردی است که مالتی تسک بودن هر نوع سیستمی را امکان پذیر می‌کند» است.

سؤال 4

پاسخ نیمه درست

نمره 0.50 از 1.00

کدام یک از جمله یا جملات زیر صحیح است؟

a. سیستم عامل یک نرم افزار مبتنی بر I/O است که با استفاده از رویکرد virtualization در مدیریت منابع، protection را هم فراهم می‌کند

b. در پیاده‌سازی Software As a Service در کلادها استفاده از VM کامل برای ارایه خدمات به کاربر مناسبتر از استفاده از کانتینر است ✖

c. در محاسبات peer to peer، تعدادی ماشین به هم متصل هستند که بعضی نقش سرور و بعضی نقش کلاینت دارند

d. از طریق جابه جایی ماشین‌های مجازی کاربران روی سرورهای مختلف کلاد، می‌توان توازن بار سرورها را کنترل و مدیریت کرد ✔

پاسخ شما تا حدودی صحیح است

You have selected too many options.

پاسخ درست «

از طریق جابه جایی ماشین‌های مجازی کاربران روی سرورهای مختلف کلاد، می‌توان توازن بار سرورها را کنترل و مدیریت کرد»

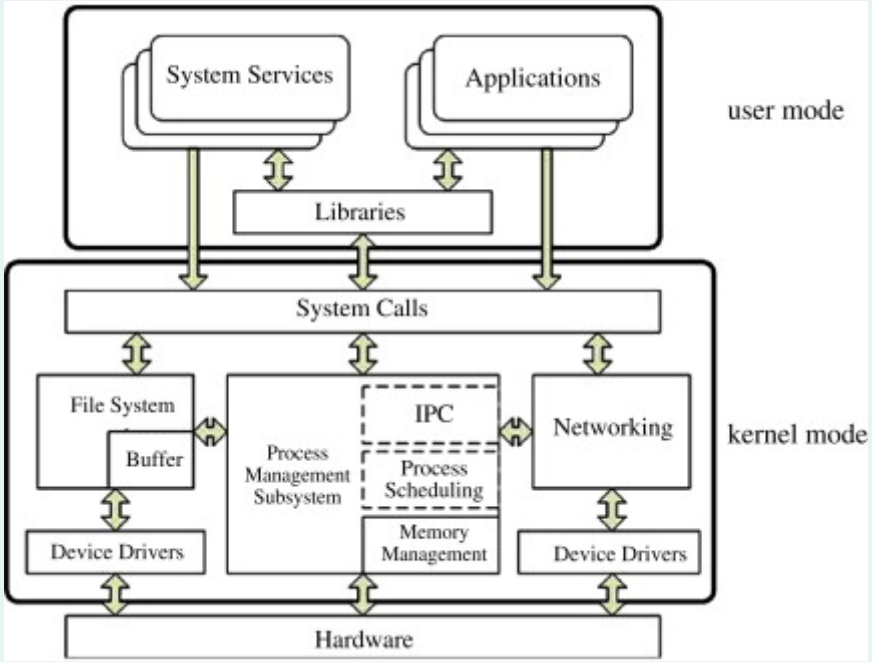
است.

سؤال 5

کامل

نمره 1.00 از 1.00

با توجه به شکل زیر، هر آنچه از رابطه دیوایس درایورها با system callها به نظرتان می آید بنویسید. منظور این است که توابع system call چه ارتباطی با دیوایس درایورها دارد . توجه کنید که تنها راه ارتباطی ما با کرنل و درخواست های ما به کرنل از طریق system callها است. پس وقتی یک درایور جدید به سیستم اضافه میکنیم چگونه از امکانات این درایور جدید استفاده خواهیم کرد؟ همچنین میتوانید به ارتباط بقیه ماژولهای کرنل مانند ماژولهای شبکه و فایل سیستم با درایورها و system call ها فکر کنید.



هر device controller ای یک سیستم عامل داره به نام device driver که مدیریتش میکند.

دیوایس درایور ها هم در بخش نرم افزاری هستند
device driver رابط ما بین سخت افزار و اپلیکیشن هاست.

مثلا یک پراسسی میخاهد یک داده ای را بخواند :

پروسه یک read call را صادر میکند و اون کال هم میاد یک سیستم کال را فراخوانی میکند.

سیستم کال هم ابتدا بافر کش را چک میکند و درستی درخواست را هم بررسی میکند.
اگر نیاز به یک io باشد که اجرا شود سیستم کال یک پیام به device driver صادر میکند. و

device driver هم یک بافر را برای خواندن allocate میکند. و عملیات IO را برنامه ریزی میکند.

در این مرحله خواندن اغاز میشود و بسته به اینکه سنکرون باشد یا نه cpu پروسه فعلی را بلاک میکند. (یا موازی پیش میرود)

device controller هم عملیات انتقال خواندن را انجام میدهد .(DMA)

وقتی عملیات کامل شد دستگاه یک سیگنال interrupt میفرستد که عملیات انجام شد.

اینتراپت هندلر هم اون پروسه ای که بلاک شده بود را به حالت اماده برش میگرداند.

در نهایت دیتاها از بافر کرنل به بافر یوزر منتقل میشوند.

سیستم کال هم به یوزر مد برمیگرد.

در نتیجه سیستم کال هایی که مربوط به read یاwrite باشند از دیوایس درایور ها هم در فرایند کامل کردن پروسه استفاده میکنند.

برای کار با فایل یا کارهای مربوط به شبکه هم میشود یک مثالی مثل مثال بالا مطرح کرد.

جهت انعطاف پذیری و امکان اضافه کردن دیوایس‌های مختلف، API ثابتی در Interface system call برای همه شامل توابع عمومی مانند read، open و write در نظر گرفته شده است . وقتی که کد درایور یک دیوایس را مینویسیم فانکشن‌هایی را برای آن بازنویسی میکنیم که هر کدام به موقع فراخوانی یک system call اجرا می‌شوند. در واقع هر دیوایس درایور با توجه به نوعش (کاراکتری یا بلاکی یا شبکه) با ماژولهای دیگری از کرنل در ارتباط است و در آخرین لایه نزدیک کاربر با interface system call در ارتباط است. مثلا نوع تعامل سیستم با تعداد زیادی دیوایسها به صورت فایل است بنابراین ماژول چنین دیوایسهایی در زیر لایه ماژول سیستم فایل قرار میگیرد و در درایور دیوایس باید توابع openو read و writeو close خاص خود آن دیوایس پیاده سازی شود. پس وقتی فایل مربوط به این دیوایس باز میشود یا خوانده یا نوشته می‌شود توابع مربوطه که در درایور پیاده سازی شده ، فراخوانی میشود.

همچنین برای دیوایسهای شبکه، ماژولهای میانی بین دیوایس شبکه و system call برای پیاده سازی استک پروتکلهای شبکه نوشته شده است و در درایور دیوایس به صورت خام داده ها از دیوایس شبکه خوانده میشود و به اولین ماژول سطح بالاتر آن تحویل داده میشود یا بالعکس از لایه های بالا دریافت و بایت بایت به بافر دیوایس شبکه ارسال می‌کند . بنابراین خود به خود system callهایی که مربوط به شبکه است به دیوایس درایور مربوطه در لایه های پایین ارجاع داده میشود

دیدگاه: