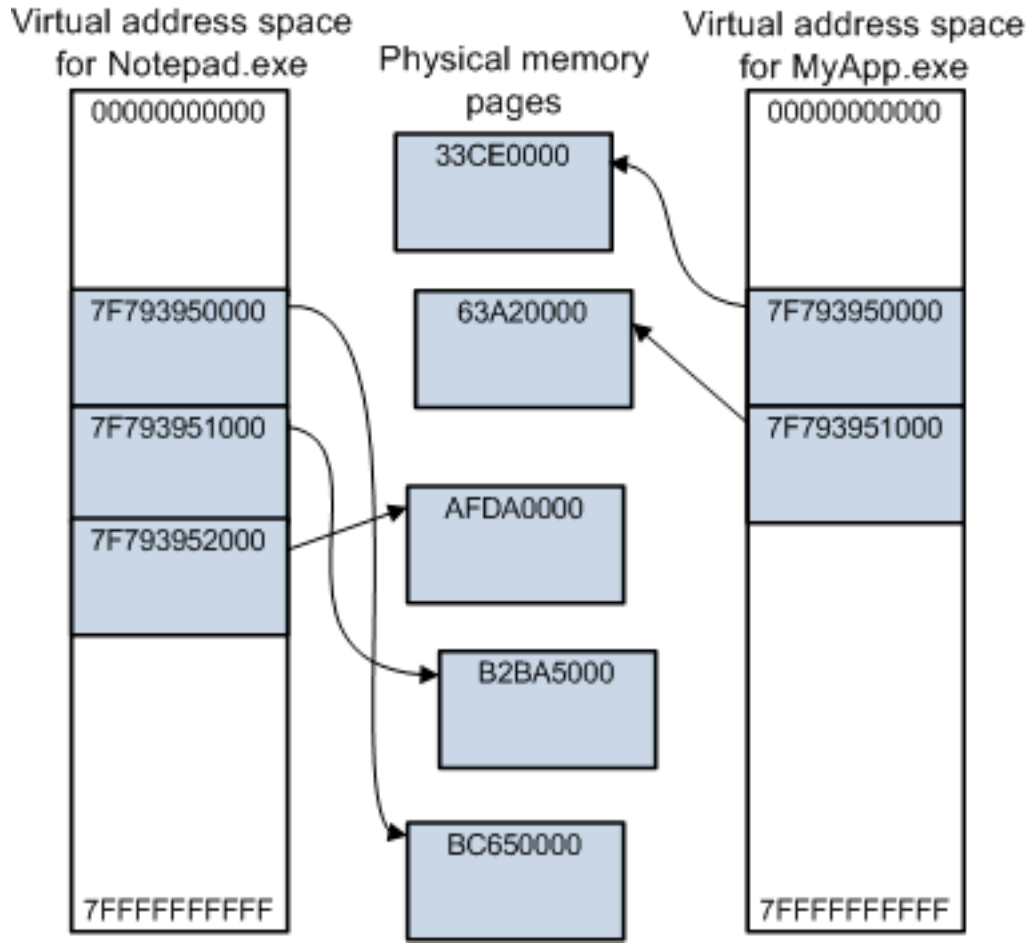
عنوان: **آشنایی با ذخیره سازی در حافظه**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۰۹/۳۰ ۱۰:۲۵
آدرس: www.dotnettips.info
گروه‌ها: IIS, Memory Manegement, Operating System

آشنایی با Virtual Address spaces

فضای آدرس‌دهی مجازی: موقعی که یک پردازشگر در مکانی از حافظه عمل خواندن و نوشتن را آغاز می‌کند، از آدرس‌های مجازی بهره می‌برد. بخشی از عملیات خواندن و نوشتن، تبدیل آدرس‌های مجازی به آدرس‌های فیزیکی در حافظه است. این عمل سه مزیت دارد:

آدرس‌های مجازی به صورت پیوسته و پشت سر هم هستند و آدرس دهی بسیار راحت است ولی داده‌ها بر روی یک حافظه به صورت متصل به هم یا پیوسته ذخیره یا خوانده نمی‌شوند و کار آدرس دهی مشکل است. پس یکی از مزایای داشتن آدرس دهی مجازی پشت سر هم قرار گرفتن آدرس هاست.
برنامه از آدرس‌های مجازی برای دسترسی به بافر حافظه استفاده می‌کند که بزرگتر از حافظه فیزیکی موجود هست. موقعی که نیاز به حافظه بیشتر باشد و حافظه سیستم کوچکتر یا کمتر از تقاضا باشد، مدیر حافظه، صفحات حافظه فیزیکی را به صورت یک فایل (عموما 4 کیلوبیی) بر روی دیسک سخت ذخیره می‌کند و صفحات داده‌ها در موقع نیاز بین حافظه فیزیکی و دیسک سخت جابجا می‌شود.
هر پردازشی که بر روی آدرس‌های مجازی کار می‌کند ایزوله شده است. یعنی یک پروسه هیچ گاه نمیتواند به آدرس‌های یک پروسه دیگر دسترسی داشته باشد و باعث تخریب داده‌های آن شود.

به محدوده شروع آدرس‌های مجازی تا پایان آن محدوده، فضای آدرس‌دهی مجازی گویند. هر پروسه ای که در مد کاربر آغاز میشود از یک فضای آدرس خصوصی یا مختص به خود استفاده می‌کند. برای سیستم‌های 32 بیتی این فضا میتواند دو گیگ باشد که از آدرس 0x00000000 شروع می‌شود و تا 0x7FFFFFFF ادامه پیدا می‌کند و برای یک سیستم 64 بیتی تا 8 ترابایت می‌باشد که از آدرس 0x000'00000000 تا آدرس 0x7FF'FFFFFFFF ادامه می‌یابد. گاهی اوقات به محدوده آدرس‌های مجازی، حافظه مجازی می‌گویند. شکل زیر اصلی‌ترین خصوصیات فضای آدرس‌های مجازی را نشان می‌دهد:



در شکل بالا دو پروسه 64 بیتی به نام‌های *notepad.exe* و *myapp.exe* قرار دارند که هر کدام فضای آدرس‌های مجازی خودشان را دارند و از آدرس 0x000'0000000 شروع و تا آدرس 0x7FF'FFFFFFFF ادامه میابند. هر قسمت شامل یک صفحه 4 کیلویی از حافظه مجازی یا فیزیکی است. به برنامه نوت‌پد دقت کنید که از سه صفحه پشت سر هم یا پیوسته تشکیل شده که آدرس شروع آن 0x7F7'93950000 می باشد ولی در حافظه فیزیکی خبری از پیوسته بودن دیده نمی‌شود و حتما این نکته را متوجه شدید که هر دو پروسه از یک آدرس شروع استفاده کرده‌اند، ولی به آدرسی متفاوت از حافظه فیزیکی نگاشت شده اند.

تفاوت user mode و kernel mode

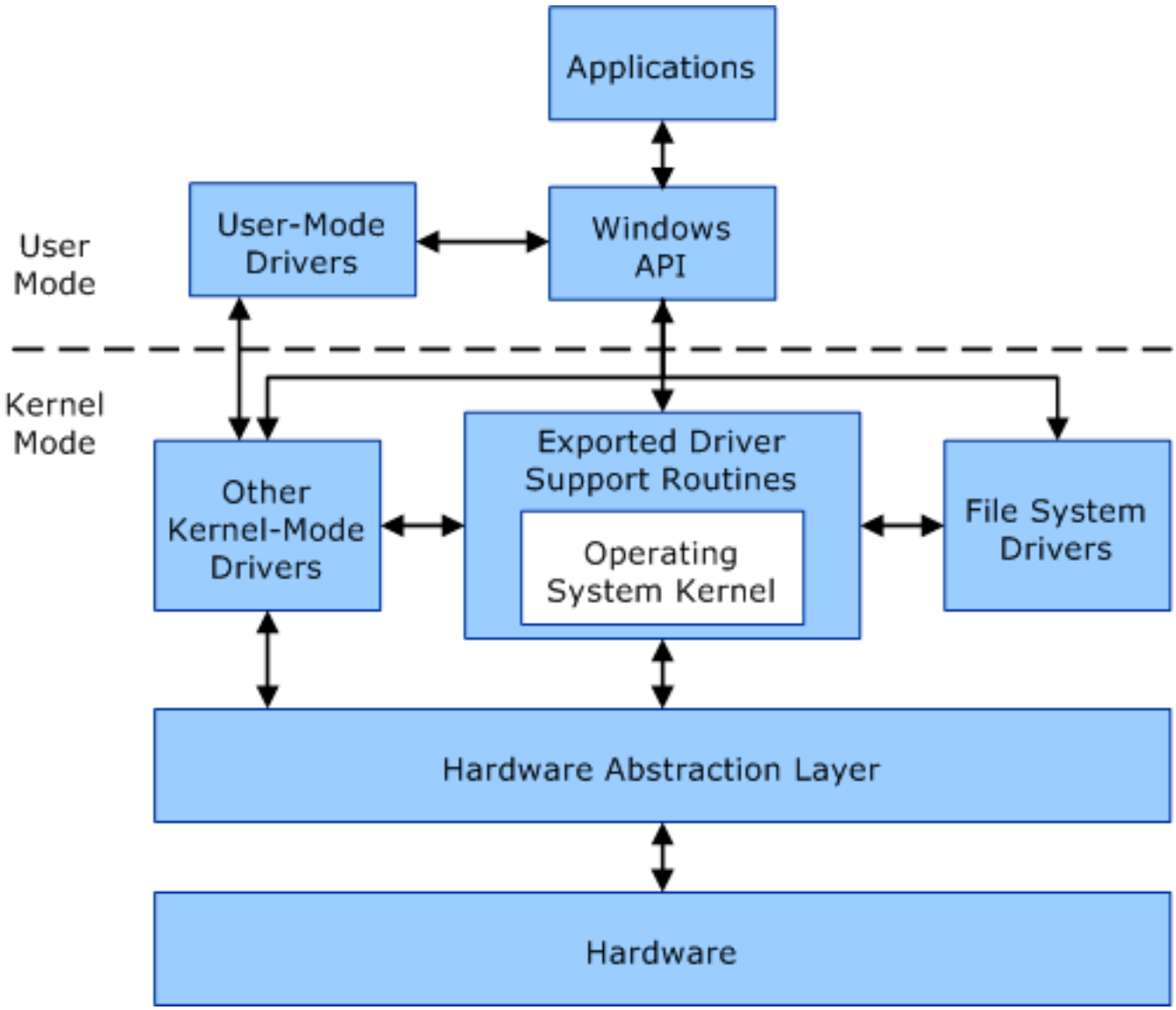
هر پردازش در سیستم بر اساس user mode مد کاربر یا kernel mode مد کرنل اجرا میشود. پردازش‌ها بر اساس هر نوع کد بین این دو بخش سوییچ می‌کنند. اپلیکیشن‌ها بر اساس مد کاربر و هسته سیستم عامل و اکثر درایورها بر اساس مد کرنل کار می‌کنند؛ ولی تعدادی از آن‌ها هم در مد کاربر.

هر برنامه یا اپلیکیشنی که اجرا می‌شود، در یک مد کاربری قرار می‌گیرد. ویندوز هم برای هر برنامه یک پروسه یا فرآیندی را ایجاد می‌کند. پروسه برای برنامه یک فضای آدرس‌دهی مجازی و یک جدول مدیریت به صورت خصوصی یا مختص همین برنامه تشکیل می‌دهد. به این ترتیب هیچ برنامه دیگری نمی‌تواند به داده‌های برنامه دیگر دسترسی داشته باشد و هر برنامه در یک محیط ایزوله شده برای خودش قرار میگیرد و این برنامه اگر به هر ترتیبی کرش کند، برنامه‌های دیگر به کار خود ادامه می‌دهند و هیچ تاثیری بر برنامه‌های دیگر نمی‌گذارند.

البته استفاده از این آدرس‌های مجازی محدودیت هایی هم دارد، چرا که بعضی از آن‌ها توسط سیستم عامل رزرو شده اند و برنامه نمی‌تواند به آن قسمت‌ها دسترسی داشته باشد و این باعث می‌شود که داده‌های برنامه از خسارت و آسیب دیدن حفظ شوند.

تمام برنامه هایی در حالت کرنل ایجاد می‌شوند، از یک فضای آدرس مجازی استفاده می‌کنند. به این معنی که یک درایور مد کرنل نسبت به دیگر درایورها و خود سیستم عامل به هیچ عنوان در یک محیط ایزوله قرار ندارد. بنابراین ممکن است یک کرنل درایور تصادفا در یک آدرس مجازی اشتباه که میتواند متعلق به سیستم عامل یا یک درایور دیگر باشد بنویسد. یعنی اگر یک درایور کرنل کراش کند کل سیستم عامل کرش میکند.

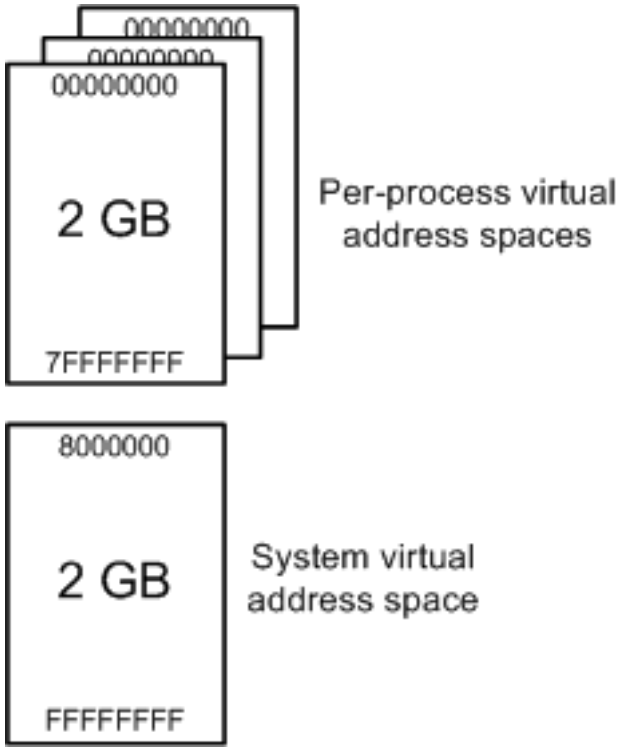
تصویر زیر به خوبی ارتباط بین مد کاربری و مد کرنل را نشان می‌دهد:



فضای کاربری و فضای سیستمی User space and system space

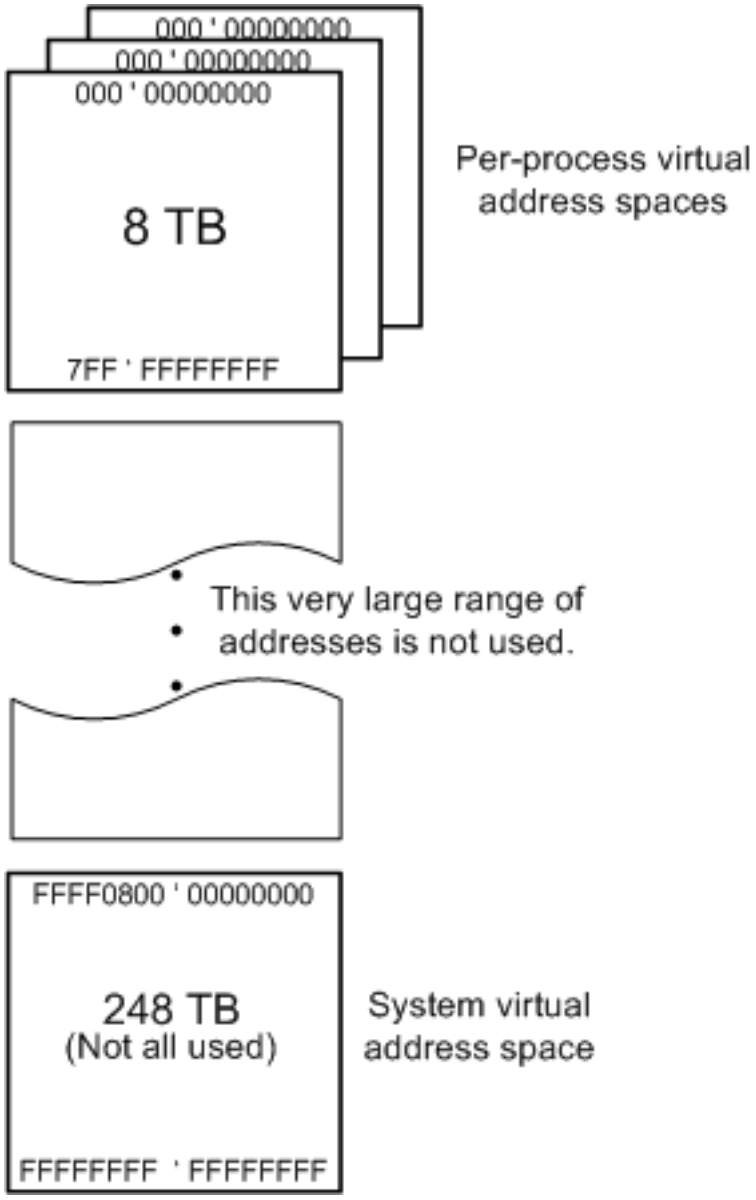
گفتیم بسیاری از پروسه‌ها در حالت user mode و پروسه‌های هسته سیستم عامل و درایورها در حالت kernel mode اجرا می‌شوند. هر پروسه مد کاربر از فضای آدرس دهی مجازی خودش استفاده می‌کند ولی در حالت کرنل همه از یک فضای آدرس دهی استفاده می‌کنند که به آن فضای سیستمی می‌گویند و برای مد کاربری می‌گویند فضای کاربری.

در سیستم‌های 32 بیتی نهایتا تا 4 گیگ حافظه می‌توان به این‌ها تخصیص داد؛ 2 گیگ ابتدایی به user space و دو گیگ بعدی به system space :



در ویندوزهای 32 بیتی شما امکان تغییر این مقدار حافظه را در میان بوت دارید و می‌توانید حافظه کاربری را تا 3 گیگ مشخص کنید و یک گیگ را برای فضای سیستمی. برای اینکار می‌توانید از برنامه `bcedit` استفاده کنید.

در سیستم‌های 64 بیتی میزان حافظه‌های مجازی به صورت تئوری تا 16 اگزابایت مشخص شده است؛ ولی در عمل تنها بخش کوچکی از آن یعنی 8 ترابایت استفاده می‌شود.



کدهایی که در user mode اجرا می‌شوند فقط به فضای کاربری دسترسی دارند و دسترسی آن‌ها به فضای سیستمی به منظور جلوگیری از تخریب داده ممکن نیست. ولی در حالت کرنل می‌توان به دو فضای سیستمی و کاربری دسترسی داشت. درایورهایی که در مدکرنل نوشته شده اند باید تمام دقت خود را در زمینه نوشتن و خواندن از فضای سیستمی در حافظه به کار گیرند.
سناریوی زیر به شما نشان می‌دهد که چرا باید مراقب بود:
برنامه جهت اجرا در مد کاربر یک درخواست را برای خواندن داده‌های یک device را آماده می‌کند. سپس برنامه آدرس شروع یک بافر را برای دریافت داده، مشخص می‌کند.
وظیفه این درایور یک قطعه در مد کرنل این است که عملیات خواندن را شروع کرده و کنترل را به درخواست کننده ارسال می‌کند.
بعد device یک وقفه را به هر تردی thread که در حال اجراست ارسال می‌کند تا بگوید، عملیات خواندن پایان یافته است. این وقفه توسط ترد درایور مربوطه دریافت می‌شود.
حالا دیگر درایور نباید داده‌ها را در همان جایی که گام اول برنامه مشخص کرده است ذخیره کند. چون این آدرس که برنامه در مد کاربری مشخص کرده است، با نمونه‌ای که این فرآیند محاسبه می‌کند متفاوت است.

Paged Pool and NonPaged Pool

در فضای کاربری تمام صفحات در صورت نیاز توانایی انتقال به دیسک سخت را دارند ولی در فضای سیستمی همه بدین صورت نیستند. فضای سیستمی دو ناحیه حافظه تخصیصی پویا دارد که به نام‌های *paged pool* و *nonpaged pool* شناخته می‌شوند.
در سیستم‌های 32 بیتی *Pagedpool* توانایی 128 گیگ فضای آدرس دهی مجازی را از آدرس 0xFFFFAC00'00000000 تا آدرس

0xFFFFAC1F'FFFFFFFF و در سیستم‌های 64 بیتی توانایی 128 گیگ فضای آدرس دهی مجازی را از 0xFFFFA800'00000000 تا 0xFFFFA81F'FFFFFFFF دارد. حافظه ای که به صورت *paged pool* تخصیص شده باشد می‌تواند صفحات حافظه را بر روی دیسک سخت ذخیره کند؛ ولی حافظه ای که به صورت *nonpaged* تخصیص یافته باشد، هرگز نمی‌تواند.