CPL/RPL/DPL

CPL：全称current privilege level，存放在代码段寄存器中（cs），代表当前执行程序的特权级

RPL: 全称request privilege level，请求特权级，存放在段选择子中

DPL: 全称descriptor privilege level，存放在段描述符中，用于表示段的特权级

CPL RPL与DPL 之间的区别和联系

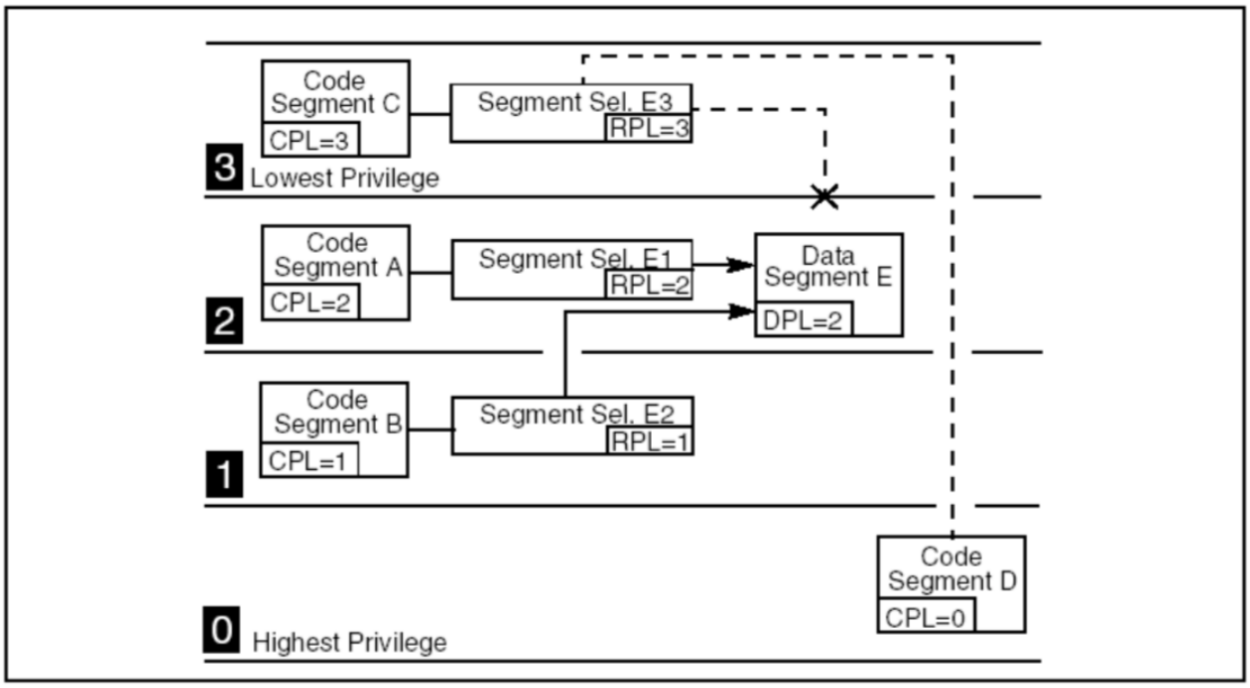


Figure 4-4. Examples of Accessing Data Segments From Various Privilege Levels

图 4—4 显示了 4 个进程（分别在代码段 A，B，C，D），每个进程都运行在不同的特权级，每个进程都试图访问同一个数据段。

- 代码段 A 的进程可以通过段选择符 E1 来访问数据段 E，因为代码段 A 的 CPL 和段选择符 E1 的 RPL 与数据段 E 的 DPL 相等。
- 代码段 B 上的进程可以通过段选择符 E2 来访问数据段 E，因为代码段 B 的 CPL 和段选择符 E2 的 RPL 都在数值上比数据段 E 的 DPL 小（也就是 CPL 和 RPL 具有更高的特权级）。B 代码段的进程也可以通过段选择符 E1 来访问数据段 E。
- 代码段 C 上的进程不能通过段选择符 E3(dotted line)，因为代码段 C 的 CPL 和段选择符 E3 的 RPL 在数值上都比数据段 E 的 DPL 大（意味着较小的特权级）。即使代码段 C 的进程使用段选择符 E1 或 E2，RPL 的够级别了，但是 CPL 的级别不够，仍然不能访问数据段 E。
- 代码段 D 上的进程本应当可以访问数据段 E，因为代码段 D 的 CPL 在数值上比数据段 E 的 DPL 在数值上更小。然而段选择符 E3 的 RPL 在数值上比数据段 E 的 DPL 大，因此该进程对数据段 E 的访问被禁止了。如果代码段 D 上的进程使用段选择符 E1 或 E2 来访问数据段，那么对数据段 E 的访问就是被允许的。

主要解决我学习中这两个大问题

- ?疑问一：RPL是用来限制什么的？
- ?疑问二：CPL保存在CS寄存器段选择子中的RPL字段中，那RPL就等于CPL吗？可是明明RPL是等于目标段的DPL啊？

我们先来明确一个检查的大体思路
在Intel开发手册卷三中，有这样写到：The processor checks the RPL along with the CPL to determine if access to a segment is allowed.

也就是说，CPU是否能访问一个段呢？其通过将RPL和CPL和在一起的特权值与段描述符中的DPL进行比较，如果DPL的值大的话，就证明可以访问该段（特权级数字越大，权利越小）

那如何来确定CPL和RPL和在一起的特权值呢？

?RPL（**Request Privilege level**）进程对段访问的请求权限

其存在于段选择子的bit 0 和 bit 1两位中

Figure 3-6. Segment Selector

段选择子
A segment selector is a 16-bit identifier for a segment . It does not point directly to the segment, but instead points to the segment descriptor that defines the segment.

IA-32中对段选择子的介绍少之又少，只是说其一个指向GDT中段描述符的一个16位的标识符。我们需要理解下面一些内容：

1. 处理器总共是提供了 6 个段寄存器来保存段选择子。
2. 当一个进程要访问某个段的时候，这个段的段选择子必须被赋值到某一个段寄存器中。因此，尽管系统定义了数千个段，只有 6 个段是可以被直接使用的。其他段

只有在他们的段选择子被置入这些寄存器中时才可以被使用。

3. 对任何程序的执行而言，至少要将代码段寄存器(CS)，数据段寄存器(DS) 和堆栈段寄存器(SS)赋予有效的段选择子。此外，处理器还提供了另外 3 个数据段寄存器(ES，FS 和 GS)供进程使用。

说了这么多只想引出来：在同一时刻程序中可以有多个段选择子，也就是可以有多个**RPL**，然而只有**CS**寄存器（也就是存放正在执行的代码的寄存器）中的**RPL**才等于**CPL**。

CS段寄存器指向的是CPU中当前运行的指令，所以CS中选择子的RPL位称为当前特权级CPL，这样的解释再合理不过了。就连IA-32这个偷懒的手册里也有这样一句话：The CPL is the privilege level of the currently executing program or task. It is stored in bits 0 and 1 of the CS and SS segment registers.

所以我们咬定无论何时CS.RPL中的存放的值就是CPL。

好了，废话说了这么多。我们就是想得出 **CS.RPL**的值 **= CPL**的值
我们暂时与RPL告一段落，理解了RPL下面两个也就比较简单了?

?CPL(Current Privilege level)当前进程的权限级别

在 CPU 中运行的是指令，其运行过程中的指令总会属于某个代码段，该代码段的特权级，也就是代码段描述符中的 DPL，便是当前 CPU 所处的特权级，这个特权级称为当前特权级，即 CPL(Current Privilege Level)。

也就是说当前你的正在运行的代码所在代码段的段描述符中的DPL等于CPL也等于CS.RPL

但是需要注意的是，我们要进行特权级判断的时候是拿目标代码段的DPL与CPL和RPL（注意这里为RPL不一定为CS.RPL，同一时刻，程序中可以有很多个段选择子，也就可以有很多个RPL）来进行判断的。
（我知道这里可能会有点绕，大伙坚持坚持 ???）

?那当前特权级为什么会改变呢?

当前正在运行的代码所在的代码段的特权级 DPL 就是处理器的当前特权级，当处理器从一个特权级 的代码段转移到另一个特权级的代码段上执行时，由于两个代码段的特权级不一样，处理器当前的特权身份起了变化，这就是当前特权级 CPL 改变的原因。

其实就是使用了那些能够改变程 序执行流的指令，如 int、 call 等，这样就使 cs 和 EIP 的值改变，从而使处理器执行到了不同特权级的代 码 。 不过，特权转移可不是随便进行的，处理器要检查特权变换的条件，这里的条件就是我们一开始说的CPL与DPL和RPL的数值比较。

?DPL(Descriptor privilege level)规定了访问该段的权限级别

IA-32中是这样说的：
The DPL is the privilege level of a segment or gate. It is stored in the DPL field of the segment or gate descriptor for the segment or gate. When the currently executing code segment attempts to access a segment or gate, the DPL of the segment or gate is compared to the CPL and RPL of the segment or gate selector

DPL是段或门的特权级别。它存储在段或门的段或门描述符的DPL字段中。当当前执行的代码段试图访问某个段或门时，将该段或门的DPL与该段或门选择器的CPL和RPL进行比较。

▲图 4-5　段描述符格式

?好我们来解决一下我们的疑问一：**RPL**是用来限制什么的？为什么我们需要它，单纯的用**DPL**和**CPL**来限制不好吗？

这个我们要先知道调用门所带来的一些系统危险⚠。

简单来说吧，调用门也就允许一段程序由低特权级变成高特权级。可以说我们平时的系统调用就是用了调用门。（这里就不多叙述了，感兴趣的推荐：操作系统真相还原一书第五章特权级那一节有详细讲解）那本来是一件好事的，但是，这样也会带来一些危险：比如某些不怀好意的人员，利用调用门将自己的权限提升到**0**级，然后对**os**内核进行破坏，这样来说危险是很大了！

那我们来分析一下为什么会产生这种情况的原因：
原因就是：受访者不知道访问者的真实身份。在受访者看来，是 0特权级的操作 系统想要数据，它还以为请求者是操作系统呢 。 实际情况是请求者为 3 特权级下的用户程序，内核程序只是代替用户程序来拿数据的。
问题就出在这，我们要想办法让受访问者知道，真正请求资源的是谁，它到底有没有资格获取这些数据。如果它有权限的话就把资源给它，否则直接拒绝请求。

而**RPL** 完美地解决了这个问题

RPL, Request Privilege Level，请求特权级，其实它代表真正请求者的特权级，也就是说，你是一个普通用户，特权级为3，当你通过调用门后，你的CPL为0，但是你的这个段选择子的RPL位仍为3，所以cpu一看，小样，你还是3级用户，我的资源可不能让你霍霍。

所以我们以后在请求某特权级为 DPL 级别的资源时，参与特权检查的不只是 CPL，还要加上 RPL. CPL 和 RPL 的特权必须同时大于等于受访者的特权 DPL，即:

数值上 **CPL <= DPL** 并且 **RPL <= DPL**

RPL 引入的目的是避免低特权级的程序访问高特权级的资源，现在的特权检查的步骤如下：

DPL 相当于权限的门槛，它代表进入本描述符所对应内存区域的最低权限，任何想迈过这个门槛的人， 它的 RPL 和 CPL 权限必须都要大于等于 DPL，即数值上 CPL <= DPL && RPL <= DPL

总结下不通过调用门、直接访问一般数据和代码时的特权检查规则

对于受访者为代码段时 ：

1. 如果目标为非一致性代码段（受到隔离的代码，只能在同一级别间相互访问），要求:数值上 CPL=RPL=目标代码段 DPL;
2. 如果目标为一致性代码段（不受到隔离的就是，允许被同等级或低等级代码调用），受访者为代码，只有在特权级转移时才会被用到，所以有关代码的特权检查都发生在能够改变代码 段寄存器 cs 和指令指针寄存器 EIP 的指令中，即这些指令要么改变 EIP，要么改变 cs 和 EIP。例如 call、 jmp, int、 ret、 sysexit 等能改变程序执行流的指令。

- • 如果目标为
- 非一致性代码段（受到隔离的代码，只能在同一级别间相互访问），
- 要求:数值上 CPL=RPL=目标代码段 DPL
- • 如果目标为
- 一致性代码段（不受到隔离的就是，允许被同等级或低等级代码调用）
- 要求:数值上(CPL >= 目标代码段 DPL && RPL >= 目标代码段 DPL)

对于受访者为数据段时：

- 要求数值上(CPL <= 目标数据段DPL && RPL <= 目标数据段 DPL)
- 栈段的特权级检查比较特殊，因为在各个特权级下，处理器都要有相应的栈，也就是说栈的特权等级要和 CPL 相同。

- 所以往段寄存器 SS 中赋予数据段选择子时，处理器要求 CPL 等于栈段 选择子对应的数据段的 DPL，即数值上 CPL=RPL =用作栈的目标数据段 DPL。

受访者若为数据，特权级检查会发生在往数据段寄存器中加载段选择子的时候，数据段寄存器包括 DS和附加段寄存器ES、 FS、 GS。

?这样看起来太完美了，完美的引出了我们的疑问二：**CPL**保存在**CS**寄存器段选择子中的**RPL**字段中，那**RPL**就等于**CPL**吗？可是明明**RPL**是等于目标段的**DPL**啊？
你上面大张旗鼓的说要用RPL与CPL一起来判断，可是你的CPL不是存放在CS寄存器的段选择子中的RPL位吗？那这两者不就是一样的了吗？

注意哦，咱们也能看出来，这里你说的是CS.RPL可不是RPL哦。
RPL和CS.RPL（也就是CPL）可不要弄混了！

不要误以为RPL和CPL都是对同一个程序而言的，它们也许不都属于同一个程序。

RPL 是位于选择子中的，所以，要看当前运行的程序在访问数据或代码时用的是谁提供的选择子，如果用的是自己提供的选择子，那肯定 CPL 和 RPL 都出自同一个程序。如果选择子是别人提供的，那就有可能 RPL 和 CPL 出自两段程序。

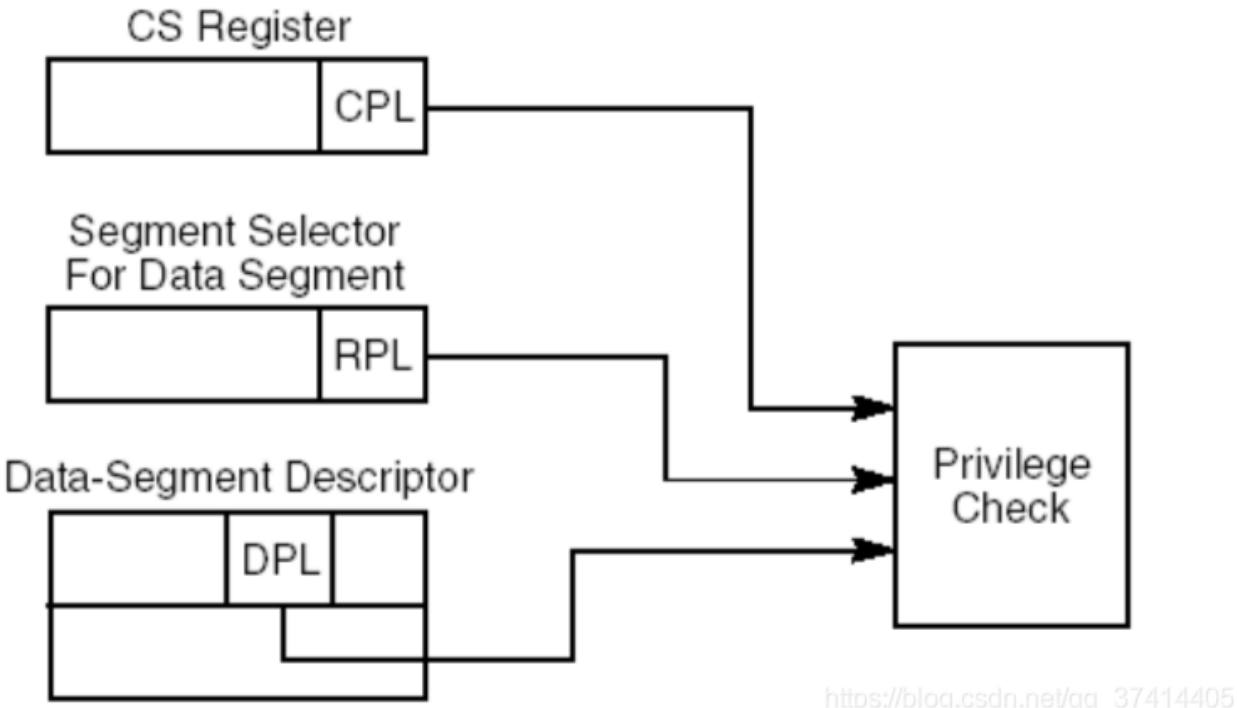
CPL 是对当前正在运行的程序而言的，而 RPL 有可能是正在运行的程序，也可能不是 。

CPL 是 指代理人，即内核， RPL 则有可能是委托者，即用户程序，也有可能是内核自己。

这个的理解话我有一个很好的例子：
访问数据段时的特权级检验

为了访问数据段中的操作数，就必须将该数据段的段选择符装载入数据段寄存器(DS，ES，FS 或 GS) 或者装载入堆栈段寄存器(SS)。(可以用如下指令装载段寄存器，MOV，POP，LDS，LES，LFS，LGS 和 LSS 指令)。

处理器将段选择符装载入段寄存器之前，它要进行特权级检验(见图)，比较当前 运行的进程或任务的特权级(CPL)，段选择符的 RPL，还有该段的段描述符的 DPL。如果 DPL 在数值上比 CPL 和 RPL 都大或者相等，处理器会将段选择符装载入段寄存器。否则，处理器会产生一个通用 保护错，并且不会装载段寄存器。



从图中就可以看出，RPL和CPL是两个不同的值

last

在文章最后，我粘一个《操作系统真相还原》里讲RPL的例子吧，觉得很形象：
不知道大伙儿学车了没有，报考驾校也要有个年龄限制，即使考 C 本 B 本也要分年龄的。假如某个 小学生 A (用户进程)特别喜欢开车，他就是想考个驾照，可驾校的门卫(调用门〉一看他年龄太小都不 让他进门，连填写报名登记表的机会都没有，怎么办?于是他就求他的长辈 B (内核〉帮他去报名，长辈 的年龄肯定够了，门卫对他放行，他来到驾校招生办公室后，对招生人员说要帮别人报名。人家招生人员 对 B 说，好吧，帮别人代报名需要出示对方的身份证(RPL)，于是长辈 B 就把小学生 A 的身份证(现在 小孩子就可以申请身份证，只是年龄越小有效期越短，因为小孩子长得快嘛)拿出来了，招生人员一看， 年纪这么小啊，不到法制学车年纪呢，拒绝接收。这时候驾校招生人员的安全意识开始泛滥了，以纵容小孩子危险驾驶为名把长辈 B 批评了一顿(引发异常)。