现的。

11.4.3 进程和线程的实现

本节将用更多细节来讲述Windows如何创建一个进程。因为Win32是最具文档化的接口，因此我们将从这里开始讲述。我们迅速进入内核来理解创建一个新进程的本地API调用是如何实现的。这里有很多细节我们都将略过，比如在创建一个路径的时候，WOW16 和WOW64 有怎样专用的代码，以及系统如何提供特定应用的修补来修正应用程序中的小的不兼容性和延迟错误。我们主要集中在创建进程时执行的主代码路径，以及看一看我们已经介绍的知识之间还欠缺的一些细节。

当用一个进程调用Win32 CreateProcess系统调用的时候，则创建一个新的进程。这种调用使用kernel32.dll中的一个（用户态）进程来分几步创建新进程，其中会使用多次系统调用和执行其他的一些操作。

1) 把可执行的文件名从一个Win32 路径名转化为一个NT 路径名。如果这个可执行文件仅有一个名字，而没有一个目录名，那么就在默认的目录里面查找（包括，但不限于，那些在PATH环境变量中的）。

2) 绑定这个创建过程的参数，并且把它们和可执行程序的完全路径名传递给本地API NtCreateUserProcess。（这个API 被增加到Window Vista使得创建进程的细节可以在内核态里处理，从而让进程可以在可信的边界内使用。之前介绍的那些API仍然是存在的，只是不再被Win32的CreateProcess调用使用。）

3) 在内核态里运行，NtCreateUserProcess执行参数，然后打开这个进程的映像，创建一个内存区对象（section object），它能够用来把程序映射到新进程的虚拟地址空间。

4) 进程管理器分配和初始化进程对象。（对于内核和执行层，这个内核数据结构就表示一个进程。）

5) 内存管理器通过分配和创建页目录及虚拟地址描述符来为新进程创建地址空间。虚拟地址描述符描述内核态部分，包括特定进程的区域，例如自映射的页目录入口可以为每一个进程在内核态使用内核虚拟地址来访问它整个页表中的物理页面。

6) 一个句柄表为新的进程所创建。所有来自于调用者并允许被继承的句柄都被复制到这个句柄表中。

7) 共享的用户页被映射，并且内存管理器初始化一个工作集的数据结构，这个数据结构是在物理内存缺少的时候用来决定哪些页可以从一个进程里面移出。可执行映像中由内存区对象表示的部分会被映射到新进程的用户态地址空间。

8) 执行体创建和初始化用户态的进程环境块(PEB)，这个PEB为用户态和内核用来维护进程范围的状态信息，例如用户态的堆指针和可加载库列表(DLL)。

9) 虚拟内存是分配在（ID表）新进程里面的，并且用于传递参数，包括环境变量和命令行。

10) 一个进程ID从特殊的句柄表（ID表）分配，这个句柄表是为了有效地定位进程和线程局部唯一的ID。

11) 一个线程对象被分配和初始化。在分配线程环境块（TEB）的同时，也分配一个用户态栈。包含了线程的为CPU寄存器保持的初始值（包括指令和栈指针）的CONTEXT记录也被初始化了。

12) 进程对象被放入进程全局列表中。进程和线程对象的句柄被分配到调用者的句柄表中。ID表会为初始线程分配一个ID。

13) NtCreateUserProcess向用户态返回新建的进程，其中包括处于就绪并被挂起的单一线程。

14) 如果NT API失败，Win32代码会查看进程是否属于另一子系统，如WOW64。或者程序可能设置为在调试状态下运行。以上特殊情况由用户态的CreateProcess代码处理。

15) 如果NtCreateUserProcess成功，还有一些操作要完成。Win32 进程必须向Win32子系统进程csrss.exe注册。Kernel32.dll向csrss.exe发送信息——新的进程及其句柄和线程句柄，从而进程可以自我复制了。进程和线程加入子系统列表中，从而它们拥有了所有Win32的进程和线程的完整列表。子系统此时就显示一个的带沙漏光标表明系统正运行，但光标还能使用。当进程首次调用GUI函数，通常是创建新窗口，光标将消失（如果没有调用到来，2秒后就会超时）。

16) 如果进程受限，如低权限的Internet Explorer，令牌会被改变，限制新进程访问对象。

17) 如果应用程序被设置成需要与当前Windows版本加垫层（shim）地兼容运行，则特定的垫层将运行（垫层通常封装库调用以稍微修改它们的行为，例如返回一个假的版本号或者延迟内存的释放）。