关于 Linux 兼容内核的知识产权问题(二)

毛德操

在本文的第一部分中，笔者就 Linux 兼容内核的开发是否会侵犯微软知识产权的问题作了些说明和讨论，结论是：只要我们小心谨慎，从总体上说 Linux 兼容内核的开发并不涉及微软的知识产权。但是，那只是就兼容内核本身而言。一个操作系统并不只是一个内核，还需要有一些运行于用户空间的系统软件的参与，才能为应用软件的运行提供必要的环境。以 Linux 为例，除内核以外至少还要有 Shell，还要有作为动态连接库(在 Unix/Linux 的术语中称为“共享对象”、即 SO)出现的 C 运行库、还要有以服务进程出现的 X11 图形界面软件，如此等等。那么，在兼容内核上面将需要有什么样的系统软件、什么样的动态连接库呢？这些软件的使用或开发会不会侵犯微软的知识产权呢？本文的第二部分就要来讨论这个问题。

从原理上说，一旦有了兼容内核，只要把 Windows 上的所有应用软件、系统软件、以及包括动态连接库在内的所有中间件通统都搬过来就是，因为这些程序在运行时所看到的将是一个 Windows 内核，或者与 Windows 等价的内核。同时，所有可动态安装的设备驱动、即.sys 模块也都可以照搬，都可以装入兼容内核运行。可是，那只是就纯技术的角度而言，这里面确实有知识产权的问题。例如，要是用户把微软的“控制面板”或者“记事本”什么的拷贝到兼容内核上，拿来就用，那显然就侵犯了微软的版权，这跟使用盗版软件没有什么不同。

所以，实际上用户空间软件的知识产权是更需要认真对待的问题。我们需要在对这些软件进行具体分析的基础上制定我们的策略。而这种分析又需要从静态和动态两个方面来进行。

先作静态分析，就是从软件的来源上进行分析。Windows 系统用户空间软件的来源大体上可以分成五类。

第一类是由第三方开发和提供的软件、包括开源软件。用户买了这些产品，或者通过别的合法手段取得了这些产品，当然有权在兼容内核上运行。只要不加以复制和分发，这里就没有知识产权的问题。

第二类是由微软开发和提供，但是作为第三方软件的配套软件、由第三方提供给最终用户。例如，第三方软件开发商买了 MFC 的开发环境(Visual Studio)，在这个环境中开发出来的应用软件的运行依赖于 MFC 的运行库。开发商可以把这个库静态地连接到其产品、即应用程序的可执行映像中，也可以采用相应的动态连接库。如果采用了动态连接库，那就应该在向用户交付其产品的同时配套提供这个动态连接库。用户既然向产品的提供商购买了产品，也就取得了对相应动态连接库的使用权。而第三方软件开发商则在向微软购买 MFC 开发环境时取得了向用户配套分发 MFC 动态连接库的权利。这里并不存在侵犯微软版权的问题。

第三类是由微软开发，并且单独作为产品出售的软件。例如微软的 Office 套件就是这样。Office 套件不是跟 Windows 操作系统捆绑销售的，用户可以单独购买。既然用户向微软购买了 Office 套件，这个特定的套件就是用户的财产，用户当然可以让它在兼容内核上运行，

这里也不存在侵权的问题。用户也许会因为某种原因(例如价格便宜)而采用由第三方开发的类似产品、例如 OpenOffice 等等，那是用户自己的选择，与兼容内核并无关系。至于 OpenOffice 等等是否涉及微软的知识产权，那是不能一概而论的，需要具体地加以分析。一般而言，应用软件的开发更容易涉及某些作为“方法”的专利，因而更需要小心从事。但是，具体的应用软件是否侵犯知识产权与兼容内核并没有什么关系。如果说兼容内核为侵权的应用软件提供了运行的平台，那么 Windows 内核同样也提供了这样的平台，因为盗版软件同样可以在 Windows 内核上运行。

第四类是由微软开发、并且只与 Windows 操作系统捆绑销售的软件。例如网络浏览器 IE 就是这样。这种捆绑销售至少是不合理的，是否违法则众说纷纭，美国与欧洲常常有针对微软捆绑销售的诉讼，也有人在推动针对此种捆绑销售的立法，因为捆绑销售已经成为垄断的一种手段。具体到中国，笔者以为也应该有人去推动反捆绑销售的立法，即使在美国一时还不能禁止捆绑销售，中国也完全可以走在前面。但是，不管怎么说，那总是需要比较长的时间。在法律还允许此种捆绑销售的时候，既然不能单独从微软购买此类软件，那就只好加以仿制。前面讲过，应用软件的仿制比较容易涉及专利，应该小心从事。不过，就 IE 而言，无论是 Mozila 还是 Firefox，都没有听说有涉及侵犯专利的嫌疑。

第五类是由微软开发、并且确实应该归入操作系统范畴、从而理应与操作系统捆绑在一起的软件。在这方面，Windows 上的一些“系统 DLL”、即系统动态连接库，就是典型的例子。特别地，Windows 上有四个动态连接库、即 DLL，起着至关重要的作用，那就是 user32.dll、gdi32.dll、kernel32.dll、以及 ntdll.dll。离开了这几个 DLL，Windows 上的应用软件就都无法运行。广义地说，介于 Win32 API 界面与内核之间的所有 DLL 都应归入此类。此外，还有些系统工具，类似于 Linux 上系统目录/bin、/sbin 中的那些工具软件，也都应归入此类。显然，属于这一类的软件为数不少。这些软件当然是不能单独向微软购买的，所以只能加以仿制。实际上，Wine 项目已经为 Linux 内核仿制了大多数重要的 Windows 系统 DLL。既然是仿制，如前所述，就要注意避免涉及专利。那么，Wine 项目所进行的仿制有没有这方面的嫌疑呢？至少笔者没有听说。这里，前几年微软对当时 Lindows 公司的诉讼可以作为一个佐证。Lidows 的产品是基于 Wine 的，但是微软告的是“Lindows”这个词与“Windows”过于接近，因而有不正当竞争之嫌，而对于 Wine 却未置一词。由此可见，微软自己也不认为当时的 Wine 有侵犯其版权或专利之嫌。

综上所述，有必要加以仿制的主要是第五类的软件、以及一些第四类的软件。要使兼容内核投入实际的使用，第五类软件是必不可少的，因而是必须要仿制的。而第四类软件，则往往是具体的应用软件。实际上，这样的软件大多已经有了仿制品，或者并非仿制但功能相近的开源或第三方产品。

这里还应指出，上述讨论不仅适用于用户空间的软件，也适用于可动态装入内核的设备驱动程序。在 Window 系统上，这就是后缀名为.sys 的设备驱动模块。同样，.sys 模块也有微软提供和第三方提供之分。由微软提供的.sys 模块理应视作 Windows 操作系统的一部分，因而都应该划入第五类。换言之，对于由微软提供的.sys 模块，如果需要的话，就得加以仿制，这样才能避开版权的问题。而既是仿制，就又要注意专利问题。

这是从静态的角度分析，再从动态的角度来分析，也就是从应用软件运行时控制流的角度来分析。

我们知道，应用软件与 Windows 操作系统之间的界面是 W32 API。应用软件一调用 W32 API 的某个函数，控制就流出了应用程序。但是，一般而言此时的控制也并没有流入内核，因为 W32 API 并不是 Windows 内核的系统调用界面。实际上，在这两个界面之间总是有一些 DLL，这些 DLL 就是所谓系统 DLL。控制必须流经某个或某些系统 DLL，才能穿过系统调用界面进入内核。有时候则并不进入内核，而只是在这些 DLL 中转了一圈就又回到了应用程序中。

如果进入了内核，那就又可以按控制流的轨迹分成两大类。

一类是在内核中转了一圈以后就按原路退出了内核，回到 DLL 中；然后又退出 DLL，回到了原先的应用程序中。在这整个过程中，控制流一直留在同一个进程中(不计与此控制流无关的进程调度)，只是在“深度”上“纵向”地流动。控制流进入内核的目的就是为了完成某些实质性的操作。大多数应用，特别是传统模式的应用，都属于这一类。显然，对于这一类的控制流，中间的 DLL 是必经之路，所以这些 DLL 都是由微软提供的系统 DLL，属于上述的第五类，这是必须加以仿制的，事实上 Wine 已经加以仿制。

另一类是在内核中转了一圈以后就主动发生进程调度和切换，按预定的要求进入了另一个进程、即“服务进程”。然后，在服务进程应邀完成某些实质性操作以后，控制流又进入内核，又发生进程调度和切换，最后回到原先的应用程序。显然，这就是 C/S 模式。在这种模式中，客户进程控制流进入内核的目的仅在于进程间通信，实质性的操作都由服务进程完成，尽管这些实质性的操作很有可能最终还是由服务进程通过某些系统调用在内核中完成。在这种模式中，控制流就不仅仅是纵向流动、而且也要作“横向”流动了。这样，不仅系统 DLL 是必经之路，服务进程也是必不可少的了。于是，服务进程所执行的软件由谁提供就是个问题。如果是由第三方软件商提供当然没有问题。而若是由微软提供，那就又要看是否与内核捆绑，如果捆绑的话就只能仿制，而仿制又要注意专利问题。

综上所述，就内核本身以外的这些软件而言，避免侵犯微软版权的手段是仿制，仿制时需要小心谨慎，以免侵犯专利、特别是“方法”专利。为此，我们需要有人专门盯住和研究美国、欧洲等主要国家和地区在计算机技术方面的专利。当然，更重要的是中国专利，因为我们将来的用户主要在中国。