高级Windows2000Rootkit检测技术(1) PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/140/2021_2022__E9_AB_98_E7_BA_A7Wind_c100_140416.htm 摘要：本文描述了一种检测内核与用户级rootkit的新技术。此技术利用处理器的单步执行模式，来测定系统内核与dll中执行指令的数量，从而达到检测rootkit和后门的目的。同时还讨论了其在win2k下的代码实现。 背景知识 pxU s9[wVYH$[本资料来源于贵州学习网 http://www.gzu521.com]pxU s9[wVYH$ 一个在计算机安全领域中重要的问题是，如何判断给定的主机是否已被入侵。由于以下两点这项工作变的非常困难： 1、攻击者可以利用未知漏洞进入系统。 2、当进入系统后，入侵者可通过安装rootkit和后门来隐藏自身(例如：隐藏进程，通讯渠道，文件等)。本文将集中讨论在win2k系统下rootkit的检测问题。 传统rootkit检测技术中的一些问题 传统的rootkit检测程序(那些我们经常在unix系统中见到的)只能检测一些已知的rootkit(这点使它变的像反病毒程序)或进行一些内核存储的扫描。例如linux中就有一些工具扫描内核中的syscall table。这显然不够好，因为已经有了很多并不更改syscall table的rootkit，而win2k下也可开发出类似的rootkit。 那检测程序是不是还应该扫描内核代码空间?这样我们就有了一个运行在内核模式中的tripwire。但这还不够好，因为在大多数的操作系统中，我们可以写出即不更改sst(syscall table)也不更改代码的内核级rootkit。在系统中有很多可以勾连的函数指针(例见[2])。以我看，存储扫描技术决不会成为rootkit检测的终结。这主要是因为我们不能确定具体的监测存储区域。 那到底怎样检

测出我们系统中的入侵者呢? 首先我们以rootkit中所使用的技术，将其分为两类： *通过更改系统结构来隐藏某对象的(如运行的进程)和 *更改内核执行路径(例：勾连那些负责枚举活动进程的内核函数)来达到同样目的的。 更改系统数据结构的rootkit 这类的rootkit不太多。有意思的例子包括fu rootkit(见[3])，通过删除内核中psactiveprocesslist链上的进程对象来隐藏进程。 zwquerysysteminformation等函数是不能发现这些隐藏进程的。但同时，因为windows的线程分派器(dispatcher， scheduler)使用另外的数据结构，这些"隐藏"进程仍可运行(被分配到cpu使用时间)。windows的线程分派器使用以下三个(注1)数据结构： *pkidispatcherreadylisthead *pkiwaitinlisthead *pkiwaitoutlisthead 后两个是处于"等待"状态的线程链头。他们之间稍有不同，但对我们来说并不重要。 从上面的信息我们可以找到一种检测隐藏进程的方法。及读取线程分派器使用的数据结构，而不是psactiveprocesslist。 当检测rootkit时我们应该尽可能的触及更底层的内核数据结构。 有一点应该注意，直接从线程分派器使用的链中删除要隐藏的进程是不可能的，因为隐藏的进程将分配不到cpu使用时间。 更改执行路径的rootkit 这类的rootkit较为普及。他们通过修改或增加内核或系统dll中的指令来达到目的。检测这类rootkit的问题是，我们不知道 rootkit在什么地方做了那些修改。它可以勾连dll中的函数，系统服务列表(system service table)，改变内核函数的内容或修改内核中一些奇怪的函数指针。 执行路径分析(execution path analysis) epa关注这样一个事实：如果入侵者通过修改执行路径隐藏了一些对象，那么当调用一些典型的系统和库的函数时，系统将运行一些多余的指令。 100Test 下