用计算机安全核心技术实现病毒防范

郝　瑾　(珠海市医疗中心电脑室　519000)
涂光辉　(珠海市公安局电脑科)

摘要:本文首先引入计算机安全核心的概念,然后介绍一种采用计算机安全核心技术设计安全卡的具体实现。

一、计算机安全核心技术

为了建立一种具有高度安全的计算机系统,我们引入计算机安全核心技术的概念。

在基本不改变计算机系统的体系结构的情况下,有许多方法可以提高系统的安全性。但对于非常敏感信息的保护,则需要一个严密的开发策略以及一个特殊的体系结构。安全核心技术就是构造安全操作系统的目前唯一最常用的技术。

安全核心的概念是 1972 年 Roger schell 首次提出,并将其定义为引用监控器(Reference monitor)的硬件和软件。安全核心技术的基础是引用控制器,它是负责实施系统安全策略的硬件和软件的组合体,引用监控器的关键是要对主体到客体的每次访问实施控制。

安全核心的理论基础是:在一个大的操作系统内,只有相对较小的一部分软件负责实施系统安全,通过对操作系统的重构,将与安全有关的软件隔离在操作系统的一个可信核内,而操作系统的大部分软件无需负责系统安全。图 1 所示是计算机系统中的安全核心。

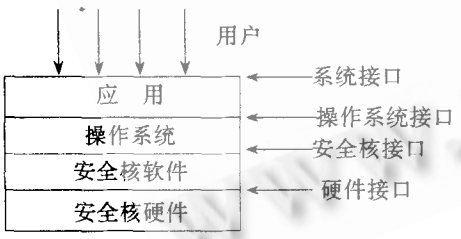

图 1　计算机系统安全核心

在一个实际应用的安全核心系统中,安全核心软件部分与操作系统无论在逻辑上还是物理上都不可能存在十分清晰的界线,安全核心软件是融合于操作系统的各个层次的,只有这样才有可能真正实现安全核心的完备性。

二、计算机安全核心的实现

这里介绍一种采用计算机安全核心技术设计的安全卡的具体实现,该项目已于 93 年 5 月通过江苏科委的鉴定。

1. 设计思想

(1)基于 DOS 操作系统。这是因为目前微机操作系统绝大多数都使用 DOS,用户面较广;且 DOS 无任何安全措施,安全性最差;微软公司将 DOS 的程序清单已全部公开。绝大多数的计算机病毒是针对 DOS 编制的,危害最大。

(2)针对病毒传播和激发的本质机理,而不是依靠病毒的表面特征进行技术处理。抛弃对各种病毒特征码逐个扫描搜索的办法,也不使用对比引导区校验的方法。在对 DOS 深入分析的基础上,从最底层入手,监测一切非授权进行写操作的渠道,巧妙地进行技术处理,瓦解病毒传播的基础。

(3)硬件加密措施。对安全控制软件的核心部分进行硬件加密,其加密控制电路为时序逻辑电路,当安全系统初始化结束之后,任何用户都不能读写此核心软件,即使熟悉计算机的人员也难以破译。

2. 安全控制系统设计

(1)病毒防范。非法进行写操作是计算机病毒最根本的特征。因此防范病毒的关键在于监视可能进行写操作的各种途径,准确地判断将要发生的写操作的合法性,拦截病毒可能侵入微机的通路。一般防病毒卡和消病毒软件都是以读写方式打开文件来判断是否对该文件有写操作。这样有可能出现"误报"和"漏报"现象。如在 DEBUG 状态下以读写方式打开一个可执行文件就报警,而实际上程序可能未执行写操作,这就出现"误报";另一情况是:如果病毒程序不通过读写方式打开一个可执行文件,而以更隐密的方式进行写操作,则一般防病毒卡和消病毒软件察觉不到。经实际测试,现有的华星卡、新创

卡、瑞星卡以及消病毒软件 CPAV 等对这种传染机制的病毒均失去效力，不能报警。本安全卡巧妙地进行了技术处理，周密地检查写操作物合法性，有效的阻止了病毒的传染。

(2)硬件加密原理。安全卡硬件系统的基本职能是为安全控制软件系统提供一个物理环境，以保证实现安全系统的病毒防范功能。为防止安全控制软件失窃，对安全控制软件的核心部分采取了硬件加密措施。硬件加密是本安全卡硬件设计的关键，主要采取两种硬件加密设计方案。

①置乱法。利用可编程逻辑阵列 GAL 系列芯片的可逻辑重构性，将地址总线上地址信息经逻辑置换后再送到安全卡上的存储芯片。微机可正确地读出安全控制程序的内容。而安全卡上 EPROM 芯片中存放的却是被置乱后的安全控制程序的代码。即使攻击者对本安全卡的 EPROM 进行反汇编，由于地址线有 15 根，按照穷举各种地址排列进行试探，反汇编次数将达 15 的幂次。加之采取了软件反跟踪措施，每一次反汇编的工作量都相当大，所以直接从 EPROM 芯片获得安全控制软件非常困难。

②封锁法。在采取了置乱法后，用户通过 CPU 仍然可以正确读取安全控制软件的内容。为了更严密的防范，还进一步采取了“封锁法”硬件加密措施。其基本思想是采用时序逻辑控制电路对安全控制软件的核心部分进行有限制的封锁。

由于在微机引导期间，用户是不可能访问安全卡中的 EPROM 的，CPU 对安全卡中的 EPROM 访问是合法的；在微机引导后，CPU 从物理电路信号难以分清访问 EPROM 的合法性。这样，我们在微机启动时开启时序逻辑控制电路，在系统初始化完成后关闭时序逻辑控制电路。在关闭期间 CPU 不可以访问核心软件。由于在安全系统初始化完成后，系统本身无需再访问核心软件，所以“封锁”对安全系统正常工作没有影响，却无法窃取安全控制软件的核心部分。

三、安全核心的软硬件实现

计算机安全卡内部逻辑结构如图 2 所示。

初始化程序：在系统启动过程中对整个安全核心系统及用户数据区、核心数据区进行必要的初始化工作；

核心代码：是整个安全卡的核心，它负责对用户的访问控制其病毒防范；

用户数据区：是一个临时存储区域，用于存放用户和系统的一部分数据及计算的中间结果；

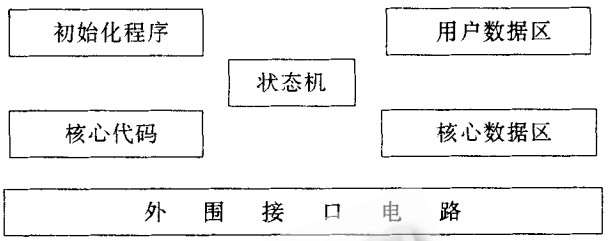

图 2　安全卡逻辑结构

核心数据区：是安全核心专用的数据区域，用户不可以直接访问的；

状态机：是为给安全核心部分提供强有力的保护，对安全核心及用户操作进行控制；

外围接口电路：是安全卡与计算机总线的接口部分，包括地址译码及数据缓冲等。

1.软件系统的结构和实现

如图 3 所示为软件系统结构：

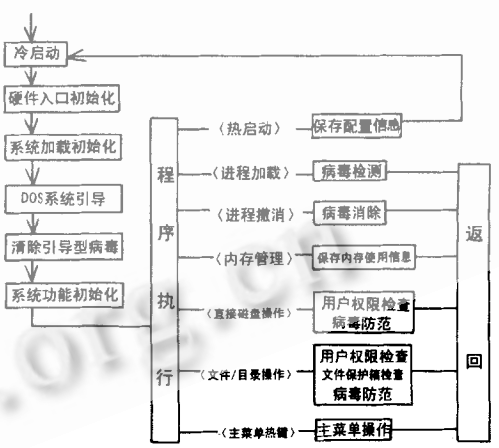

图 3　安全卡系统软件结构

(1)初始化过程

·硬件入口初始化

·系统加载初始化

·系统功能初始化

(2)磁盘操作检查

·INT 13H 中断检查

·INT 40H 检查

·INT 21H 的 44H 功能检查

(3)文件/目录操作检查

·目录项操作的检查

·文件写操作的检查
·文件信息标志的检查
(4)进程管理
·进程加载管理
·进程撤消管理
(5)内存管理

通过监视应用程调用 INT 21H 的 48H、49H、4AH 功能，保存各进程使用内存的情况。

2.硬件系统的结构和实现

本安全卡的硬件加密机制由可编程逻辑阵列和状态机实现。

逻辑阵列为一 GAL 系列芯片，内部包含一系列地址译码和控制译码，由控制信号控制地址译码部分使之指向不同的地址空间，控制信号来自状态机。状态机是一时序逻辑控制电路，用于判断 CPU 是否处于启动过程。如图 4 所示为硬件系统结构：

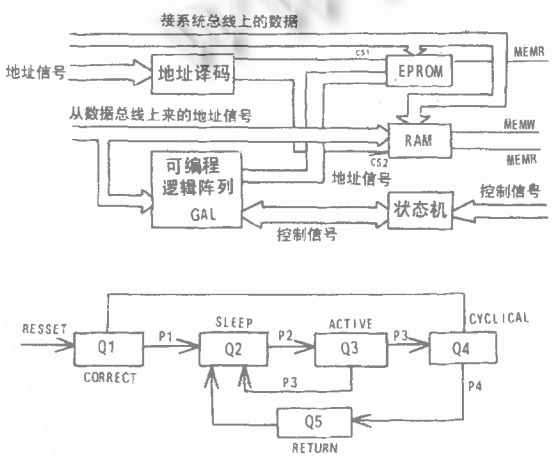

图 4　安全卡硬件结构

现设定状态由四个 D 触发器的四人输出端(Q1、Q2、Q3、Q4)经组合后共同决定。这样状态机就具有 2=16 个不同的状态，即 0000—1111，其中只有一个状态能产生正确的地址译码控制信号。状态的产生取决于总线上的控制信号和逻辑阵列的反馈信号，只有当总线上的控制信号和反馈信号正确时才获得正确状态。

该状态机具有五种状态：休眠(sleep)、激活(active)、循环(cyclical)、复位(return)及正常(correct)，在前四种状态时，状态机皆不能产生正确的控制信号，只有当 correct 状态时输出的信号才是正确的。

一般情况下状态机处于 sleep 状态，条件 A 来到时才变为 active，接下来状态机本身对 B、C、D 条件是否来到进行自我判断，一旦满足马上自动返回 sleep 状态，考虑到用户的随机性和可能的攻击性，该状态机选用了一组独特的条件信号(A、B、C、D)，这些信号用户在正常情况下不会激活的，保证只有在允许的情况下，CPU 才可以对加密部分正常读写。

这里我们选取 RESET 信号为时序逻辑控制电路的开启信号，设定状态为 q1、q2、q3、q4(Q1 表示 q1、q2、q3、q4 的一种组合)。

当 RESET 来到时状态机打开，初始状态为 Q1，这时 DOS 处于启动过程，硬件加密机制自动进行监测，在安全系统初始化完成后，立即通过硬件产生一组指令序列(p1)，将状态机复位到休眠状态(sleep)，此时状态为 Q2。

要将逻辑控制门再次打开，必须按顺序送出若干组特殊的指令序列 p2、p3、……，当 p2 来到时，状态机被激活，处于 Q3 状态，Q3 为中间状态，若下一指令序列非 p3，则状态立即返回 Q2(sleep)；若为 p3，则状态机进到 Q4 状态，Q4 仍为中间状态，若下一指令序列为 p4，则状态机进入到复位状态(return)。该状态非常短暂，状态机很快又回到 sleep 状态，若下一指令不是 p4 而是 p5，则状态机进到正常状态(correct)。此时状态机才处于正确的输出状态，给出正确的 = 控制信号，以使 CPU 能正常读写硬件加密区。

图 5　GAL 运行过程

可编程逻辑阵列的动作过程是：通过 GAL 芯片实现的可编逻辑阵列主要作用是地址译码和产生反馈信号提供给状态机，如图 5 所示：

当系统运行时，地址总线送来的信号经状态机的译码控制信号的控制后产生一组译码的地址信号，再送往 ROM 区进行选址，同时根据状态机的控制信号产生一组反馈信号给状态机，使状态要发生相应的状态转换，根据状态机输出的控制信号，逻辑地址与物理地址经过某种特定的转换后，才是 EPROM 的实际物理地址。

(来稿时间：1996 年 6 月)