[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/42 (2006.01)
G06F 12/14 (2006.01)

[12] 发明专利说明书

专利号 ZL 99811943.1

[45] 授权公告日 2006 年 1 月 18 日

[11] 授权公告号 CN 1237444C

[22] 申请日 1999.10.1 [21] 申请号 99811943.1
[30] 优先权
[32] 1998.10.9 [33] DE [31] 19846676.5
[86] 国际申请 PCT/DE1999/003169 1999.10.1
[87] 国际公布 WO2000/022506 德 2000.4.20
[85] 进入国家阶段日期 2001.4.9
[71] 专利权人 因芬尼昂技术股份公司
地址 德国慕尼黑
[72] 发明人 C·梅 J·弗雷瓦尔德
O·布里克瑟尔
审查员 袁丽颖
[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 程天正 张志醒

权利要求书 1 页 说明书 3 页

[54] 发明名称

保护入口地址的方法

[57] 摘要

一种高效地保护计算机程序中的入口地址的方法，其中通过在直接位于所述入口地址之前或之后的存储单元中存放相关数据的地址，而利用不在同一单个指令中的数据的相关性来识别允许的入口地址；或者通过用非预备存储区内的程序数据实现相关，而利用不在同一单个指令中的数据的相关性来识别允许的入口地址。 在此，可以由编译程序或链接程序通过组织程序代码来确保只有合法的入口地址才满足所述的相关性。 根据本发明无须执行所计算的跳跃，由此在 CPU 指令流水线中不会出现窥孔。

1. 保护与被调用功能相关的入口地址的方法，其中，通过在直接位于所述入口地址之前或之后的存储单元中存放相关数据的地址，而利用不在同一单个指令中的数据的相关性来识别允许的入口地址。

2. 如权利要求 1 所述的方法，其中，在所述直接位于入口地址之前或之后的存储单元中存放一个针对合法入口地址的保护目录内的项的参考。

3. 如权利要求 1 或 2 所述的方法，其中，直接跳至所述允许的入口地址。

4. 如权利要求 1～2 中任一项所述的方法，其中，在执行功能调用时，自动地为有关的入口地址检验是否满足数据的相关性。

5. 保护与被调用功能相关的入口地址的方法，其中，通过用非预备存储区内的程序数据实现相关，而利用不在同一单个指令中的数据的相关性来识别允许的入口地址。

6. 如权利要求 5 所述的方法，其中，程序指令不超过某个最大数目 n 的字节，设定一种特殊的无操作代码，用以避免随机的相关性。

7. 如权利要求 5 或 6 所述的方法，其中，在至少相距每个程序指令的最大数目字节的代码数据之间产生所述的相关性。

8. 如权利要求 5～6 中任一项所述的方法，其中，通过加入一个特殊的字节序列来暂存所述的入口地址，而该字节序列可以不出现在常规代码内。

9. 如权利要求 8 所述的方法，其中，采用一种特殊的无操作代码作为特殊的字节序列。

10. 如权利要求 5～6 中任一项所述的方法，其中，直接跳至所述允许的入口地址。

保护入口地址的方法

技术领域

本发明涉及用于保护与被调用功能相关的入口地址的方法。

背景技术

在未来的芯片卡上，应允许可以调用操作系统功能的第三方的应用。为此可能存在如下危险，即该第三方的应用包含关于其它程序段的干扰或破坏企图。这种可能的侵害有：从所述的应用出发不是采用操作系统程序的有效功能指针，而是采用另一入口地址。由此，操作系统的代码不是按规定执行的，并且可能譬如因错误重写存储器区而引起数据损失。

根据现有技术设计了一种解决方案，它利用的是进入到固定间隔（门）中的预定地址上。在此，首先必须检验门地址是否位于模块所允许的地址范围之内。在结果肯定的情况下，便跳向该门，然后从那里进一步跳跃到原本的功能中。这是有缺陷的，因为此处必须执行所计算出的跳跃，该跳跃在 CPU 流水线中将导致一个窥孔。也可选择在功能的一些预先处理的程序指令之后延迟地跳跃到所述的功能中。如果在所述功能的开始处引入一个循环，并且必须持续地在门处的代码和调出的代码之间来回跳跃，那么这将会给编译程序带来很困难的优化问题。

发明内容

因此，本发明任务在于，创造一种保护这种类型的入口地址的方法，其中无须执行所计算的跳跃，由此在 CPU 指令流水线中不会出现窥孔。

根据本发明，该任务通过如下方式解决，通过在直接位于所述入口地址之前或之后的存储单元中存放相关数据的地址，而利用不在同一单个指令中的数据的相关性来识别允许的入口地址。或者通过如下方式解决，通过用非预备存储区内的程序数据实现相关，而利用不在同一单个指令中的数据的相关性来识别允许的入口地址。

在此，编译程序或链接程序通过组织程序代码来确保只有合法的入口地址才满足该相关性。譬如可以通过如下方式来实现所述的相关

性，即直接位于入口地址之前或之后的存储单元包含有相关数据的地址。

为此，一种优选的可能性在于，直接位于入口地址之前或之后的存储单元包含有一个针对合法入口地址的保护目录内的相应项的参考。

在此，尤其优选的是，在执行功能调用时自动地检验是否满足数据的相关性。

特别优选的是，在执行功能调用时自动地检验所述的相关数据是否位于预定的预备存储区。

只要程序指令不超过某个最大数目 n 的字节，则可以通过如下方式应用本发明的另一解决方案，即设定一种特殊的无操作代码，用以避免随机的相关性，并且可以由编译程序或链接程序补充加入该无操作代码。

在此，如下做法是尤其优选的，即在至少相距 n 字节的代码数据之间产生所述的相关性。

此外，根据本发明，所述的入口地址可以通过加入一个特殊的、可以不出现在常规代码内的字节序列来进行暂存，譬如利用特殊的无操作代码。

因此，根据本发明，可以通过直接跳至功能指针的地址来避免流水线中的窥孔。但是，必须负责通过非局部的代码相关性来标出合法的跳跃地址。所述的编译程序或链接程序必须通过组织程序代码来确保只有合法的功能跳跃地址才能满足该相关性。在此，“非局部的相关性”意味着那些可以不位于同一单个指令内的数据的相关性。

因此，譬如下列的优选实施方案是可能的：

1. 利用为此而预备的存储区内的数据的相关：一种简单的实现方法可以是譬如在直接位于入口地址之前的存储单元内包含相关数据的地址，该相关数据譬如又对应着功能的合法入口地址。在执行功能调 用时，可以自动地检验是否满足该相关性，以及/或者该相关数据是否位于预定的预备存储区。该机理乍一看来非常近似于迄今的门机理，但有个优点，就是不会出现所计算的跳跃，而且可以直接把功能的指令取至流水线的预取器中。窥孔只有在非法进入的故障情况下才出现。

2. 利用非预备的存储区内的程序数据的相关。该解决方案的前提条

件是，程序指令不超过某个最大数目 n 的字节。于是必须排除代码段中有较长的数据区。此外，该方法还有一个前提条件是特殊的无操作代码（SNOP），该无操作代码从不装入在常规代码中，而只是为了避免随机的相关性而由编译程序/链接程序补充加入。在此还可以区别为两种不同的解决方案：

a) 所述的相关性是在至少相距 n 个字节的代码数据之间产生的。在此，所述的编译程序或链接程序必须确保通过引入 SNOP 中间代码来避免代码中可能的随机相关性。

一种可能的实现规定如下：直接在所述功能的入口地址之前存在一个值，该值是接下来 n + m（$m \geq 0$，否则是任意的）个字节的功能。如果代码中任何地方都可能随机地满足该相关性，则编译程序或链接程序必须消除该随机的相关性：因为根据前提条件在 n + m 个字节的序列中是以至少一个实指令结束，所以在该指令结束后可以加入一系列 SNOP 指令，直到所述的功能值发生变化。在此，该功能可以在某个界限内自由地进行选择。

b) 所述的入口地址是通过加入特殊的字节序列来进行暂存的，该特殊字节序列可以不出现在常规代码之内。此处的例子为 OP 代码序列（SNOP）。

因此，根据本发明可以通过非局部的只可能在入口地址处出现的代码相关性来暂存功能的入口地址。

由此有利地避免了所述的门机理，该门机理会产生一种计算的跳跃，而且在指令流水线中引起一种窥孔。

更具体地讲，可直接在入口地址处跳至所述的功能中。接下来的指令可被装入到流水线内，且与验证跳跃地址的对或错无关。由此提高了监视功能调用的效率。