(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113687968 B
(45)授权公告日 2024.11.26

(21)申请号 202110835650.8

(22)申请日 2021.07.23

(65)同一申请的已公布的文献号
申请公布号 CN 113687968 A

(43)申请公布日 2021.11.23

(73)专利权人 浙江众合科技股份有限公司
地址 310052 浙江省杭州市滨江区江汉路1785号双城国际4号楼17层

(72)发明人 严光明 胡明明 陈小杰

(74)专利代理机构 杭州华鼎知识产权代理事务所(普通合伙) 33217
专利代理师 戴俊波

(51)Int.Cl.
G06F 11/07(2006.01)

(56)对比文件
CN 103294517 A,2013.09.11
CN 106547606 A,2017.03.29

审查员 张露

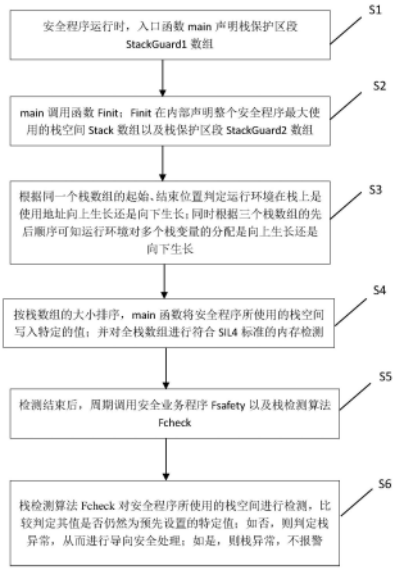
权利要求书1页 说明书3页 附图1页

(54)发明名称

一种适用通用操作系统SIL4栈检测方法

(57)摘要

本发明公开了一种适用通用操作系统SIL4栈检测方法,包括如下步骤:步骤S1、安全程序运行时,入口函数main声明栈保护区段StackGuard1数组;步骤S2、main调用函数Finit;Finit在内部声明整个安全程序最大使用的栈空间Stack数组以及栈保护区段StackGuard2数组;步骤S3、根据同一个栈数组的起始、结束位置判定运行环境在栈上是使用地址向上生长还是向下生长;步骤S4、按栈数组的大小排序,main函数将安全程序所使用的栈空间写入特定的值并对全栈数组进行内存检测;步骤S5、检测结束后,周期调用安全业务程序Fsafety以及栈检测算法Fcheck;步骤S6、栈检测算法Fcheck对安全程序所使用的栈空间进行检测。栈检测方法通用性强,动态适应各种CPU、操作系统、编译器,从而实现SIL4级栈内存检测。



CN 113687968 B

1.一种适用通用操作系统SIL4栈检测方法,其特征在于:包括如下步骤:

步骤S1、安全程序运行时,入口函数main声明栈保护区段StackGuard1数组;

步骤S2、main调用函数Finit;Finit在内部声明整个安全程序最大使用的栈空间Stack数组以及栈保护区段StackGuard2数组;

步骤S3、根据同一个栈数组的起始、结束位置判定运行环境在栈上是使用地址向上生长还是向下生长;同时根据三个栈数组的先后顺序获知运行环境对多个栈变量的分配是向上生长还是向下生长;

步骤S4、按栈数组的大小排序,main函数将安全程序所使用的栈空间写入特定的值;并对全栈数组进行符合SIL4标准的内存检测;

步骤S5、检测结束后,周期调用安全业务程序Fsafety以及栈检测算法Fcheck;

步骤S6、栈检测算法Fcheck对安全程序所使用的栈空间进行检测,比较判定其值是否仍然为预先设置的特定值;如否,则判定栈异常,从而进行导向安全处理;如是,则栈异常,不报警;

按栈数组的大小排序表示为: P1<P2<P3<P4<P5<P6,其中 区段[P1,P2]和[P5,P6]为安全程序所使用的栈空间,区段[P3,P4]为栈保护的栈空间,P1记录Stack数组的起始位置,P2记录Stack数组结束位置,P3记录StackGuard1数组的起始位置,P4记录StackGuard1数组结束位置,P5记录StackGuard2数组起始位置,P6记录StackGuard2数组结束位置。

2.根据权利要求1所述的一种适用通用操作系统SIL4栈检测方法,其特征在于:

步骤S1还包括:利用全局变量P3记录StackGuard1数组的起始位置以及利用全局变量P4记录StackGuard1数组结束位置。

3.根据权利要求1或2所述的一种适用通用操作系统SIL4栈检测方法,其特征在于:

步骤S2还包括:利用全局变量P1记录Stack数组的起始位置以及利用全局变量P2记录Stack数组结束位置,然后返回。

4.根据权利要求3所述的一种适用通用操作系统SIL4栈检测方法,其特征在于:

步骤S2还包括:利用全局变量P5记录StackGuard2数组起始位置以及利用全局变量P6记录StackGuard2数组结束位置,然后返回。

5.根据权利要求1或2所述的一种适用通用操作系统SIL4栈检测方法,其特征在于:

同一个栈数组为Stack数组、StackGuard1数组或者StackGuard2数组三者任一一个。

6.根据权利要求1所述的一种适用通用操作系统SIL4栈检测方法,其特征在于:

Fsafety所使用的栈与Finit声明的Stack数组为相同的内存空间。

一种适用通用操作系统SIL4栈检测方法

技术领域

[0001] 本发明涉及计算机数据结构技术领域,具体的,涉及一种适用通用操作系统SIL4栈检测方法。

背景技术

[0002] 栈作为一种数据结构,是一种只能在一端进行插入和删除操作的特殊线性表。它按照后进先出的原则存储数据,先进入的数据被压入栈底,最后的数据在栈顶,需要读数据的时候从栈顶开始弹出数据,最后一个数据被第一个读出来。栈具有记忆作用,对栈的插入与删除操作中,不需要改变栈底指针;一些嵌入式操作系统提供了固定栈空间大小,栈检测的功能,栈溢出作为栈检测的一项重要检测项,栈溢出是由于C语言系列没有内置检查机制来确保复制到缓冲区的数据不得大于缓冲区的大小,因此当这个数据足够大的时候,将会溢出缓冲区的范围;在系统出现异常时,无接口通知SIL4安全程序;对于闭源的操作系统,无法感知其处理是否符合SIL4要求;另外一些操作系统无栈检测功能,在有足够的资源下,可以任意分配随意多的栈空间,然而SIL4安全程序无法感知到栈是否出现越界等异常。

发明内容

[0003] 针对上述弊端,本发明的目的是提出一种适用通用操作系统SIL4栈检测方法,通过栈只往一个方向生长以及预先分配的事实,动态适应各种CPU、操作系统、编译器,从而实现SIL4级栈内存检测。

[0004] 为实现上述技术目的,本发明提供的一种技术方案是,一种适用通用操作系统SIL4栈检测方法,包括如下步骤:

[0005] 步骤S1、安全程序运行时,入口函数main声明栈保护区段StackGuard1数组;

[0006] 步骤S2、main调用函数Finit;Finit在内部声明整个安全程序最大使用的栈空间Stack数组,以及栈保护区段StackGuard2数组;

[0007] 步骤S3、根据同一个栈数组的起始、结束位置判定运行环境在栈上是使用地址向上生长还是向下生长;同时根据三个栈数组的先后顺序获知对多个栈变量的分配是向上生长还是向下生长;

[0008] 步骤S4、按栈数组的大小排序,main函数将安全程序所使用的栈空间写入特定的值;并对全栈数组进行符合SIL4标准的内存检测;

[0009] 步骤S5、检测结束后,周期调用安全业务程序Fsafety以及栈检测算法Fcheck;

[0010] 步骤S6、栈检测算法Fcheck对安全程序所使用的栈空间进行检测,比较判定其值是否仍然为预先设置的特定值;如否,则判定栈异常,从而进行导向安全处理;如是,则栈异常,不报警。

[0011] 作为优选,步骤S1还包括:利用全局变量P3记录StackGuard1数组的起始位置以及利用全局变量P4记录StackGuard1数组结束位置。

[0012] 作为优选,步骤S2还包括:利用全局变量P1记录Stack数组的起始位置以及利用全

局变量P2记录Stack数组结束位置,然后返回。

[0013] 作为优选,步骤S2还包括:利用全局变量P5记录StackGuard2数组起始位置以及利用全局变量P6记录StackGuard2数组结束位置,然后返回。

[0014] 作为优选,同一个栈数组为Stack数组、StackGuard1数组或者StackGuard2数组三者任一一个。

[0015] 作为优选,Fsafety所使用的栈与Finit声明的Stack数组为相同的内存空间。

[0016] 作为优选,按栈数组的大小排序表示为:P1<P2<P3<P4<P5<P6,其中区段[P1,P2]和[P5,P6]为安全程序所使用的栈空间,区段[P3,P4]为栈保护的栈空间。

[0017] 本发明的有益效果:本发明一种适用通用操作系统SIL4栈检测方法的通用性强,不依赖于安全程序所运行的操作系统,不依赖于目标CPU,也不依赖于所使用的编译器对栈生长方向的分配。SIL4安全程序能够按需调整栈保护的空间大小,进行特定值选择,通过栈检测算法Fcheck对安全程序所使用的栈空间进行检测,通过比较特征值的异同判定异常情况。

附图说明

[0018] 图1为本发明的一种适用通用操作系统SIL4栈检测方法的流程图。

具体实施方式

[0019] 为使本发明的目的、技术方案以及优点更加清楚明白,下面结合附图和实施例对本发明作进一步详细说明,应当理解的是,此处所描述的具体实施方式仅是本发明的一种最佳实施例,仅用以解释本发明,并不限定本发明的保护范围,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本发明保护的范围。

[0020] 实施例:如图1所示,一种适用通用操作系统SIL4栈检测方法的流程图,包括如下步骤;

[0021] 步骤S1、安全程序运行时,入口函数main声明栈保护区段StackGuard1数组;利用全局变量P3记录StackGuard1数组的起始位置以及利用全局变量P4记录StackGuard1数组结束位置。

[0022] 步骤S2、main调用函数Finit;Finit在内部声明整个安全程序最大使用的栈空间Stack数组以及栈保护区段StackGuard2数组;其中,利用全局变量P1记录Stack数组的起始位置以及利用全局变量P2记录Stack数组结束位置,然后返回;利用全局变量P5记录StackGuard2数组起始位置以及利用全局变量P6记录StackGuard2数组结束位置,然后返回。

[0023] 步骤S3、根据同一个栈数组(Stack数组、StackGuard1数组或者StackGuard2数组三者任一一个)的起始、结束位置判定运行环境在栈上是使用地址向上生长(内存地址变大)还是向下生长(内存地址变小);同时根据三个栈数组的先后顺序获知运行环境对多个栈变量的分配是向上生长(内存地址变大)还是向下生长(内存地址变小)。

[0024] 步骤S4、按栈数组的大小排序,表示为:P1<P2<P3<P4<P5<P6,其中区段[P1,P2]和[P5,P6]为安全程序所使用的栈空间,区段[P3,P4]为栈保护的栈空间;main函数将安全程序所使用的栈空间写入特定的值;并对全栈数组进行符合SIL4标准的内存检测。

[0025] 步骤S5、检测结束后,周期调用安全业务程序Fsafety以及栈检测算法Fcheck;Fsafety所使用的栈与Finit声明的Stack数组为相同的内存空间。

[0026] 步骤S6、栈检测算法Fcheck对安全程序所使用的栈空间进行检测,比较判定其值是否仍然为预先设置的特定值;如否,则判定栈异常(栈溢出或者被OS损坏),从而进行导向安全处理;如是,则栈异常,不报警。

[0027] 以上所述之具体实施方式为本发明一种适用通用操作系统SIL4栈检测方法的较佳实施方式,并非以此限定本发明的具体实施范围,本发明的范围包括并不限于本具体实施方式,凡依照本发明之形状、结构所作的等效变化均在本发明的保护范围内。

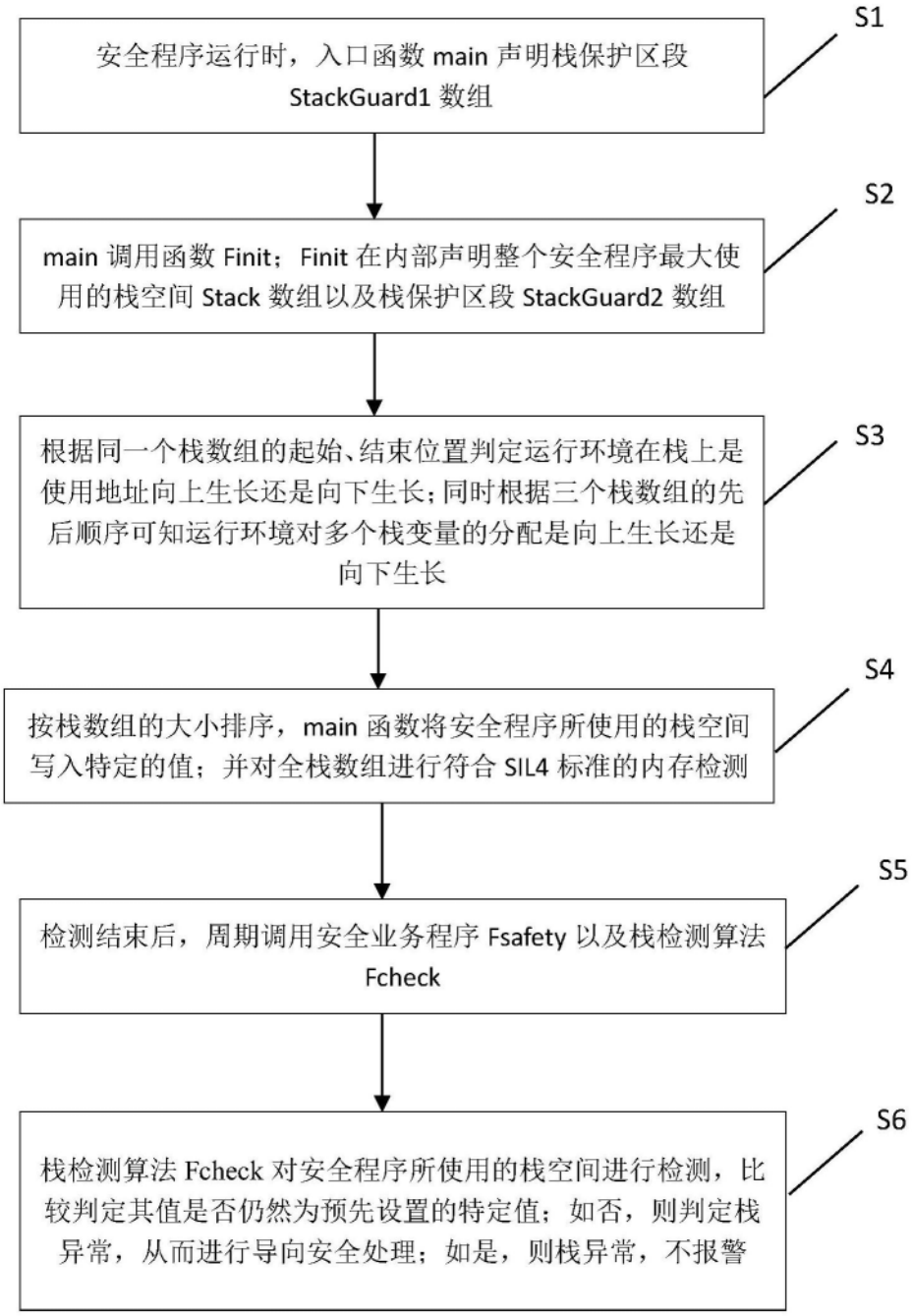

S1
安全程序运行时，入口函数 main 声明栈保护区段 StackGuard1 数组
S2
main 调用函数 Finit；Finit 在内部声明整个安全程序最大使用的栈空间 Stack 数组以及栈保护区段 StackGuard2 数组
S3
根据同一个栈数组的起始、结束位置判定运行环境在栈上是使用地址向上生长还是向下生长；同时根据三个栈数组的先后顺序可知运行环境对多个栈变量的分配是向上生长还是向下生长
S4
按栈数组的大小排序，main 函数将安全程序所使用的栈空间写入特定的值；并对全栈数组进行符合 SIL4 标准的内存检测
S5
检测结束后，周期调用安全业务程序 Fsafety 以及栈检测算法 Fcheck
S6
栈检测算法 Fcheck 对安全程序所使用的栈空间进行检测，比较判定其值是否仍然为预先设置的特定值；如否，则判定栈异常，从而进行导向安全处理；如是，则栈异常，不报警

图1