(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 112840341 B
(45)授权公告日 2024.03.15

(21)申请号 201980066605.1
(22)申请日 2019.10.10
(65)同一申请的已公布的文献号
申请公布号 CN 112840341 A
(43)申请公布日 2021.05.25
(30)优先权数据
A50889/2018 2018.10.11 AT
(85)PCT国际申请进入国家阶段日
2021.04.09
(86)PCT国际申请的申请数据
PCT/AT2019/060339 2019.10.10
(87)PCT国际申请的公布数据
WO2020/073072 DE 2020.04.16
(73)专利权人 AVL李斯特有限公司
地址 奥地利格拉茨
(72)发明人 P·普利勒
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 董华林
(51)Int.Cl.
G06F 21/56(2006.01)
G06F 9/455(2006.01)
G05B 17/02(2006.01)

(56)对比文件
CN 108255711 A,2018.07.06
CN 1711525 A,2005.12.21
JP 2005301981 A,2005.10.27
JP 2013242633 A,2013.12.05
JP 2014225160 A,2014.12.04
US 2013139262 A1,2013.05.30
US 2016085567 A1,2016.03.24
US 2018032760 A1,2018.02.01
US 8127360 B1,2012.02.28
WO 2016141998 A1,2016.09.15

审查员 彭玢

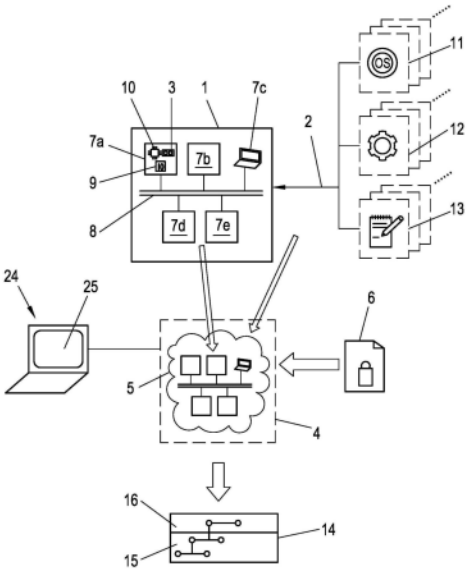
权利要求书2页 说明书11页 附图5页

(54)发明名称

用于探测与安全相关的数据流的方法

(57)摘要

用于探测与安全相关的数据流的方法，所述数据流在硬件系统(1)中在执行至少一个数据处理任务(2)期间出现。所述方法具有以下步骤：定义关键数据(6)，所述关键数据能够存储在所述硬件系统的至少一个存储器单元(3)中。将所述硬件系统(1)映射到能在仿真环境(4)中运行的仿真模型(5)上。在所述仿真环境(4)中利用所述仿真模型(5)执行数据处理任务(2)作为仿真。在执行所述数据处理任务(2)期间，监控在所述仿真模型(5)中的所述关键数据(6)和所述关键数据(6)的实例(6'、6")的创建、传输和删除。识别和记录与安全相关的数据流。



CN 112840341 B

1.用于探测与安全相关的数据流的方法,所述数据流在硬件系统(1)中在执行至少一个数据处理任务(2)期间出现,所述方法的特征在于以下步骤:

为了将所述硬件系统(1)映射到仿真模型(5)中,将所述硬件系统(1)划分为多个硬件组件(7),所述多个硬件组件经由至少一个通信装置(8)相互连接并且能够经由所述通信装置(8)交换数据,其中,每个硬件组件(7)具有至少一个通信单元(9)、至少一个存储器单元(3)和至少一个处理器单元(10),

将所述硬件系统(1)映射成能在仿真环境(4)中运行的仿真模型(5),并且利用所述硬件系统(1)的仿真模型(5)对在执行数据处理任务(2)期间出现的数据流进行仿真,其中,在仿真环境(4)中的仿真期间观察和记录由各个硬件组件(7)执行的数据流,

数据流描述:在同一存储器单元(3)或另一存储器单元(3)的另一存储地址处创建存储在所述存储器单元(3)的存储地址处的数据的拷贝;改变数据或数据的拷贝;删除存储在存储器单元(3)的存储地址处的数据;通过由多个硬件组件使用的通信装置(8)将数据从一个硬件组件(7)传输至一个或多个另外的硬件组件(7);或者经由硬件系统(1)的外部接口来传输和接收数据,

其中,定义关键数据(6),所述关键数据能够存储在硬件系统(1)的至少一个存储器单元(3)中和/或所述关键数据能够经由接口转移到所述硬件系统(1)中,

其特征在于,

将所述硬件系统的部段划分到允许域(15)和禁止域(16)中,

至少将以下数据流识别为与安全相关的数据流:

-经由禁止域(16)的通信装置(8)传输关键数据(6)或所述关键数据的实例(6’、6”),

-在所述关键数据(6)的实例(6’、6”)继续存在期间删除关键数据(6),

-在所述禁止域(16)中创建关键数据(6)的实例(6’、6”),

-在确定的时刻和/或系统状态之后关键数据(6)继续存在,

并且在执行所述数据处理任务(2)期间,监控在所述仿真模型(5)中的所述关键数据(6)和所述关键数据(6)的实例(6’、6”)的创建、传输和删除,并且识别和记录与安全相关的数据流。

2.按照权利要求1所述的方法,其特征在于,所述关键数据(6)的定义通过定义数据属性来进行。

3.按照权利要求1或2所述的方法,其特征在于,在识别到至少确定的与安全相关的数据流时创建警告通知。

4.按照权利要求1或2所述的方法,其特征在于,将所识别的与安全相关的数据流存储在分析结果(14)中。

5.按照权利要求4所述的方法,其特征在于,以图形方式表示所述分析结果(14)。

6.按照权利要求1或2所述的方法,其特征在于,记录所述仿真的流程。

7.按照权利要求1或2所述的方法,其特征在于,从对应用软件(12)的执行中得出所述数据处理任务(2)。

8.按照权利要求1或2所述的方法,其特征在于,将以下数据流识别为与安全相关的数据流:在数据处理任务完成之后关键数据(6)继续存在。

9.按照权利要求7所述的方法,其特征在于,所述应用软件在处理定义的测试用例的情

况下运行。

用于探测与安全相关的数据流的方法

技术领域

[0001] 本发明涉及一种用于探测与安全相关的数据流的方法,所述数据流在硬件系统中在执行至少一个数据处理任务期间出现。

背景技术

[0002] 在信息和通信技术(IKT)的系统中的典型任务是防止未授权的用户或应用访问信息(数据)。为此目的,在现有技术中实现不同的方法,其中,包括例如访问控制(也以英语术语称为“访问权限(Access-Rights)”或“访问策略(Access-Policies)”)、管理访问授权(“授权(Authorization)”)和隔离过程和/或存储空间(例如沙盒(Sandboxing))。在较低层级上,这例如可以在硬件中经由存储器管理单元(“Memory-Management-Units”-MMU)来实现。

[0003] 由于构思错误或实现错误、对授权的有针对性地攻击、经由旁路(“旁道(Side-Channels)”)的无意识的信息流以及类似的机制,可能得出对数据的非期望地访问的可能性(“漏洞(Leaks)”)。于是,攻击(“进攻(Attack)”)利用一个或多个薄弱环节(“弱点(Vulnerabilities“)”),以便以确定的结果(“冲击(Impact)”)获得对IKT系统的影响和/或对数据的访问。由此,可以袭击或破坏系统的机密规则(“保密性(Confidentiality)”)。

[0004] 硬件或计算机系统的保密性特性主要是通过遵循在设计和实现以及运行方面的原则和规定来满足,例如:

[0005] -每个设计的安全性(Security-Per-Design),

[0006] -按照经证明的成功模型的方式(“最佳实践(Best Practice)”),

[0007] -遵循过程或标准(例如ISO/IEC 27001和27002,ISO 15408/通用标准(Common Criteria),NIST CSF,SAE J3061),

[0008] -原则、例如在内存中加密(In-Memory-Encryption),

[0009] -地址空间布局随机化(Address Space Layout Randomization,ASLR),

[0010] 等。

[0011] 随着开发,并且尤其是最终以及在每个集成步骤之后,需要对实现和因此所产生的行为进行测试。这可以例如通过对软件和硬件的架构与实现进行审查来完成(审核(Audits))。这种方式是有效的并且自数十年以来已经得到使用,但是也是非常(在时间上)耗费的并且难以自动化的。但是尤其是,必须使系统的源(代码、模型)可供使用。这例如出于保密的原因,不总是期望的并且经常也根本不可能(例如在使用第三方的软件库时)。

[0012] 其他可能性来自静态代码分析(这是可自动化的,但在查找漏洞中不是非常有效的)或模型检查(这又要求公开源)。所述方法的共同点也在于,无法识别对处理链(包括传输、存储和处理)和其中的薄弱环节的攻击。这例如涉及编译、链接、传输(下载等)、存储(例如在闪存、硬盘等上)、在软件(解释器、虚拟机等)中的执行、以及最后的硬件。出于该原因,许多成功的攻击恰好在此着手:

[0013] 例如,Lipp M.,Gruss D.,Schwarz M.,Bidner D.,Maurice C.,Mangard S.(2017

年9月)在文章“Practical Keystroke Timing Attacks in Sandboxed Javascript)”(发表在:Focky S.,Goldmann D.,Snekkenes E.(eds)计算机安全(Computer Security)-ESORICS 2017,在计算机科学方面的讲义(Lecture Notes in Computer Science),10493卷)中,描述了经由沙盒的攻击。

[0014] 例如,Gruss,D.,Maurice,C.,Wagner,K.,&Mangard,S.(2016年7月)在“Flush+Flush:a fast and stealthy cache attack”(发表在:关于探测入侵和恶意软件以及漏洞评估的国际会议(International Conference on Detection of Intrusions and Malware,and Vulnerability Assessment)(279-299页),斯普林格出版社)中,描述了缓存攻击。

[0015] Gruss,D.,Maurice,C.,&Mangard,S.(2016年7月)在“Rowhammer.js:A remote software-induced fault attack in javascript)”(发表在:关于探测入侵和恶意软件以及漏洞评估的国际会议(300-321页),斯普林格出版社)中,描述了针对存储器的攻击。

[0016] 除了静态测试之外,也存在动态测试,所述动态测试在执行期间检查系统行为。动态测试补充静态测试、尤其是用于对应于未知或新模式(静态地尚无法搜索到的模式)的场景,以及用于在分析中由于复杂性或未考虑的特性无法静态地发现的情况。如果无法访问完整的系统描述和源代码,则静态测试的可能性无论如何受到严格限制,这可以进一步增加动态测试的重要性。

[0017] 动态测试方法通常基于(手动或自动地)执行定义的应用场景(“测试用例(Test Cases)”)。在此,挑战尤其是在于:

[0018] (i)在适合地设计所述测试用例方面,以便注意到安全问题,

[0019] (ii)在执行方面(理想地,所述执行自动化地进行,以便能够在给定的时间内执行尽可能大数量的测试情形),和

[0020] (iii)在对测试结果的正确分析方面、尤其是对安全薄弱环节的识别。

[0021] 与本发明相关联地,优选地将如下预先规定称为“测试用例”,所述预先规定定义预先条件、输入、执行条件、测试方法和/或确定的测试或确定的仿真的预期结果。

[0022] J.,Hu,W.,Irturk,A.,Tiwari,M.,Sherwood,T.,&Kastner,R.在文章“Information flow Isolation in I2C and USB)”(2011年6月,发表在:“第48届设计自动化会议论文集(Proceedings of the 48th Design Automation Conference)(254-259页),ACM)中,描述了一种经特定修改的硬件、例如在I2C外设构件中、在寄存器和内部总线中对每个数据字扩展了例如一个附加位。所述位(在文章中称为“污点(taint)”)标记“关键”信息并且通过硬件带着所述信息“迁移”。数据的处理步骤(例如逻辑链接)被扩展了所述标记位,因此可以表示并且继续跟踪所述关键数据对结果的影响。然而,在没有经特定修改的硬件的情况下,所述方法无法实现。

发明内容

[0023] 本发明尤其是具有如下目的,即,能在没有硬件改变的情况下实现动态测试,其中,该方法不仅应当能够用于标准硬件,而且尤其是也能够在工业自动化领域中用于嵌入式系统、例如在汽车领域中使用。

[0024] 按照本发明,所述目的和另外的目的通过开头所述类型的方法来实现,所述方法

具有以下步骤:定义关键数据,所述关键数据能够存储在硬件系统的至少一个存储器单元中,或者经由接口(Interface)转移到所述硬件系统中;将所述硬件系统映射到能在仿真环境中运行的仿真模型上;在所述仿真环境中利用所述仿真模型执行数据处理任务作为仿真;在执行所述数据处理任务期间,监控在所述仿真模型中的所述关键数据和所述关键数据的实例的创建、传输和删除;识别和记录与安全相关的数据流。借助于所述方法能够识别非期望的数据流在所述硬件系统中的出现,而不必为此修改所述硬件系统。尤其是,在基于标准CPU(例如Intel、ARM、Infineon等的CPU)的当前的微控制器系统中,几乎不存在获得这样经修改的硬件的可能性;但是,对于几乎所有的这种CPU,存在可以用作仿真环境的适合的仿真器。

[0025] 与本发明相关联地,作为"仿真环境"尤其是表示具有在其上运行的仿真软件的计算机系统,所述仿真软件适合于仿真模型的仿真。不同于真正的硬件系统,所述仿真环境允许监控例如存在于存储器和寄存器中的数据的尤其是当前的状态以及改变所述数据的所有过程。由此,在仿真环境中尤其是可以监控数据的创建、传输和删除。适合于对确定的硬件系统的仿真模型进行仿真的相应的仿真环境是本领域技术人员已知的,并且在获知在此所公开的教导的情况下,本领域技术人员将能够选择用于预定的硬件系统的适合的仿真环境并且将所述硬件系统映射到可由所选择的仿真环境执行的仿真模型中。

[0026] 与本发明的说明书相关联地,"实例"是表示关键数据的拷贝,所述拷贝至少在确定的时刻(例如在形成时刻/在实例化时)直接与所考察的关键数据相关。因此,实例例如可以是关键数据的拷贝,但也可以是数据的拷贝和(重新)编码。例如,如果原始数据是整数并且实例是其浮点代表,则进行重新编码。

[0027] 与本公开的技术方案相关联地,"硬件系统"通常是表示数据处理系统及其外围的物理组件(亦即、电子的和机械的组成部分)的整体。尤其是,所述硬件系统可以是按照已知系统架构的计算机系统。然而,硬件系统也可以是多个硬件组件的配合作用的结构,所述硬件组件在狭义上不对应于"计算机系统"、例如具有多个硬件组件的传感器网络或Ad-Hoc网络。

[0028] 以有利的方式,为了将所述硬件系统映射到仿真模型上,可以将所述硬件系统划分为多个硬件组件,所述多个硬件组件经由至少一个通信装置相互连接并且能够经由所述通信装置交换数据,其中,每个硬件组件具有至少一个通信单元、至少一个存储器单元和至少一个处理器单元。通过适合地选择形成用于仿真模型的建模的基础的划分,可以在不同的层级上、例如在抽象的机器或虚拟机的层级上、在CPU架构层级上、在门或位层级上、在通信或总线层级等上仿真和观察所述硬件系统。

[0029] 在一种有利的实施方式中,所述关键数据的定义可以通过定义数据属性来进行。与本公开的技术方案相关连地,数据属性尤其是理解为对于如下内容的属性:数据内容、形成或创建时刻、处理时刻、在存储器中的地址、与系统状态的相关性、与其他(关键)数据的相关性等以及由此可定义的组合。这例如允许在对软件组件的源代码没有认识的情况下进行检验。

[0030] 对于数据属性的相关组合的一个示例是观察运行时间、例如来自调用栈数据的形成和删除时刻的运行时间。一种旁道攻击例如使用关于密钥检验流程的运行时间的信息,以便间接地获取关于密钥的信息,参见Kocher,P.C.,Timing attacks on implementation

of Diffie-Hellman,RSA,DSS,and other Systems(1996年8月,发表在:年度国际密码学会议(Annual International Cryptology Conference)(104-113页),施普林格出版社,柏林,海德堡)。

[0031] 以有利的方式,可以在识别到至少确定的与安全相关的数据流时创建警告通知。所述警告通知可以立即在控制和监控系统的用于控制仿真环境的用户界面上显示,使得测试员可以决定他是中止还是继续运行所述仿真。必要时,可以在出现警告信号的情况下自动地中止所述仿真,如果这是非期望的话。

[0032] 以有利的方式,可以将所识别的与安全相关的数据流存储在分析结果中,其中,所述分析结果必要时以图形方式表示。这允许记录和报告所执行的测试。

[0033] 以有利的方式,也可以记录所述仿真的流程。此外,必要时,所包括的流程(包括历史)可以一起记录并且必要时与数据流的记录相关联,这支持后面的分析/诊断/调试。

[0034] 以有利的方式,所述数据处理任务可以从对应用软件的执行中得出,所述应用软件必要时在处理定义的测试用例的情况下运行。这允许部分地或全部地测试应用软件,并获得关于应用软件与位于所述应用软件之下的软件部分(中间件、框架、库、操作系统等)和给定的硬件系统的交互的认识。

[0035] 在按照本发明的方法另一种有利的实施方式中,所述硬件系统的部段可以被划分到允许域和禁止域中。这允许提高匹配相关性。

[0036] 到允许域和禁止域中的所述划分例如可以在关键数据的定义过程中、在仿真模型的建模过程中亦或还在后面进行,其中,所述划分尤其是或者可以基于程序员或测试员的经验来确定,或者可以基于由类似的硬件系统较早执行的测试方法来预定。基于在此公开的教导的认知,本领域技术人员能够进行相应地划分。

[0037] 以有利的方式,至少以下数据流可以作为与安全相关的数据流来识别:经由禁止域的通信装置传输关键数据或所述关键数据的实例,在所述关键数据的至少一个实例继续存在期间删除关键数据,在所述禁止域中创建关键数据的实例,在确定的时刻和/或系统状态之后、尤其是在完成数据处理任务之后或者在终止应用之后关键数据继续存在。必要时,可以通过观察所述流程来考虑附加的规则、例如与时间、系统状态的相关性等。(这基本上意味着:与仿真流程相关地改变所述允许域和禁止域的定义。)

[0038] 与本发明相关联地,“与安全相关的数据流”例如表示如下数据流,所述数据流对应于确定的、对于相应的应用情形定义的标准、例如在上面的段落中所解释的标准之一。

附图说明

[0039] 以下,参考附图1至5更详细地阐述本发明,所述附图示例性、示意性并且非限制性地示出本发明有利的实施方式。图中:

[0040] 图1示出在按照本发明的方法中所涉及的单元和方法步骤的示意图,

[0041] 图2示出硬件系统和在其上运行的软件的示意图,

[0042] 图3示出仿真环境的示意图,在所述仿真环境中,图2的硬件系统被映射为仿真模型,

[0043] 图4示出在仿真环境中执行的仿真的示意图,并且

[0044] 图5示出仿真的图形化处理的结果的图表。

具体实施方式

[0045] 在图1中抽象地示出了按照本发明的方法的重要的实体。硬件系统1通常包括多个硬件组件7(在图1中,为了区分而以小写字母a至e对所述硬件组件进行补充),所述硬件组件经由至少一个通信装置8相互连接并且能够经由所述通信装置8交换数据。为此目的,每个硬件组件7常见地具有通信单元9、存储器单元3和处理器单元10,如这在图1中象征性地对于硬件组件7a所示出的那样。在图1中所示出的硬件组件7c例如可以是常规的(笔记本)计算机,在所述计算机上例如运行用于另外的硬件组件的控制软件。硬件系统1也可以具有“嵌套的”硬件结构,亦即、所述硬件系统是上级的硬件系统的一部分。硬件系统1(或者硬件系统1的硬件组件7)例如也可以实施为SoC(片上系统)。例如对于这样的硬件系统包括例如Infineon的XC27x8X系列的微处理器。在此,通信装置8通过内部总线系统构成,通信单元9可以在CPU的总线耦合部中可见,内部寄存器和缓存可以视作存储器单元3并且处理器单元10是SoC的CPU。必要时,外围组件、例如CAN控制器、PWM控制器或者其他组件可以定义为硬件系统1的另外的硬件组件7。

[0046] 按照表示层级,硬件系统1的硬件组件7可以包括不同类型的单元。对于硬件组件7的另外的示例包括:

[0047] -车辆中的控制器(例如用于马达、变速器、ABS、ESP等的ECU或其他控制单元),

[0048] -传感器网络中、例如在IoT系统中的智能传感器,所述传感器网络经由WSN(无线传感器网络)进行通信,

[0049] -在存储器编程的(工业)控制装置(SPS)中的CPU模块和/或I/O模块,

[0050] -在云或计算机群集中的节点。

[0051] 例如,与常规的PC相关联地,主CPU可以定义为硬件组件7,其中,外围控制器、例如磁盘、键盘、显卡等可以定义为另外的硬件组件。

[0052] 通信装置8可以是内部或外部总线系统,或者其他有线或无线通信装置8。对于通信装置8的示例包括设备内部的总线、例如PCI、PCI-X、PCIe、AGP、USB、I2C等,以及外部的或者说现场总线、例如CAN、LIN、Flexray、(汽车)以太网、MOST等。在车辆领域中,这通常称为“车载网络-IVN”

[0053] 与在图1中所示出的形式不同,硬件系统1也可以包括硬件组件7和通信系统8的不同布置结构,例如硬件系统1可以具有较复杂的联网,其中,例如包括多个分等级的总线、星形或网状网络等。对于硬件系统1的在实践中相关的示例是嵌入式系统。

[0054] 与本公开的技术方案相关联地,“嵌入式系统”表示在技术背景中提及的硬件系统1。嵌入式系统例如在医疗技术设备中、在家用电器(例如洗衣机或冰箱)中、在消费电子产品(例如电视机、DVD播放器、机顶盒、路由器或移动电话)中或者不同的道路车辆、轨道车辆或水上交通工具(例如机动车、飞机或轮船)、其他运输设备、在工业设备中、但也在宇宙航行中得以应用。

[0055] 在复杂的整体系统的情况下(亦即、在复杂的技术背景中),嵌入式系统可以是多个在其他方面自主的、嵌入式的(子)系统的联网,所述(子)系统经由不同的通信装置8、例如具有分别不同的安全性规格、数据吞吐量规格和/或速度规格的多个总线系统而相互联网。

[0056] 可选地,硬件系统1和/或单独的或所有的包含在所述硬件系统中的硬件组件7具

有包括运行时间环境、运行时间库、驱动程序等的操作系统11,所述操作系统以已知的方式能实现处理数据处理任务2或者参与所述数据处理任务的处理。例如,操作系统11可以是常规的或专业特定的计算机操作系统,所述操作系统控制计算机的内部的和外部的硬件的运行和配合作用。操作系统11的示例包括在个人计算机、平板电脑和智能电话的领域中的Windows、Android、iOS、macOS、Linux、UNIX;在汽车领域或不同制造商的专营产品中的QNX、Elektrobit OS、ROS、AUTOSAR。根据硬件系统1的表示的细节程度也可以将单独的操作系统11与多个硬件组件7相关联。这于是例如是如下情况,即,如果硬件系统1是(或包括)计算机系统,则所述计算机系统的内部硬件组件7受控地由操作系统经由相应的总线系统相互通信。然而,根据系统界限和建模,也可以将“较小的”、“较大的”或“嵌套的”单元定义为硬件组件7,其中,在这种情况下将相应的规则框架理解为确定所述单元的功能性的操作系统。

[0057] 因此,与本发明的说明书相关联地,将程序指令的集合理解为操作系统11,所述集合控制硬件组件7和/或硬件组件7的组和/或整体硬件系统1的系统资源的功能或者控制在其中运行的数据流。在一个硬件系统1中也可以设置多个操作系统11。

[0058] 即使硬件系统1的表示的细节程度本身没有改变,要指出的是,在硬件组件7中的这种表示或划分对于建模的稍后描述的步骤、亦即创建硬件系统1的仿真模型5是重要的。

[0059] 与参数数据13有关地,从应用软件12中得出不同的数据处理任务2,所述数据处理任务由硬件系统1或各个硬件组件7来处理。与本发明相关联地,可以将由一个硬件组件7或者由多个硬件组件7共同执行的每个过程步骤(或者过程步骤的每个组)视为“数据处理任务”2。尤其是,与本公开的技术方案相关联地,将在其中发生数据流的过程步骤视为数据处理任务。

[0060] 与本公开的技术方案相关联地,“数据流”尤其是表示:

[0061] -在同一存储器单元3或另一存储器单元3的另一存储地址处创建存储在所述存储器单元3的存储地址处的数据的拷贝,

[0062] -改变数据或数据的拷贝,

[0063] -删除存储在存储器单元3的存储地址处的数据,

[0064] -通过由多个硬件组件使用的通信装置8将数据从一个硬件组件7传输至一个(或多个)另外的硬件组件7,以及

[0065] -经由硬件系统1的外部接口来传输和接收数据。

[0066] 与本发明的说明书相关联地,将所有在程序外部的影响因子称为“参数数据”13,所述影响因子影响应用软件12的流程。参数数据13可以涉及由开发者或用户选择的设置、或者可以从确定的环境条件中得出、或者给定典型地在配置文件(有时称为“INI”或“CFG”文件)中定义的基本设置、默认值、操作模式等。这样的参数数据13的另一示例是传感器数据,所述传感器数据在硬件系统1(所述硬件系统例如可以是在车辆中的嵌入式系统)中按照应用软件12来处理。因此,在真实的硬件系统1中,所述参数数据13由当前的环境条件得出,在对硬件系统1进行仿真的过程中,相应的参数数据13可以作为测试用例来提供和处理。

[0067] 通过叠加地表示用于操作系统11、应用软件12和参数数据13的符号应当使得可见的是,多个操作系统和/或应用软件单元和/或参数数据组对于硬件系统11可以是相关的。

[0068] 由于硬件系统1和相关联的一个操作系统11或相配的多个操作系统11的复杂架

构,并且由于利用现代编程工具所创建的应用软件12的结构,借助于静态方法不再可能的是,根据应用软件12的源代码完全地预测在确定的硬件系统1中哪些数据流将触发数据处理任务。所述检验也可以仅根据有限数量的示例性的参数数据13来进行。但是,这恰好例如可以通过经由旁路的无意的信息流开辟可能用于攻击的薄弱环节。

[0069] 但是不可能的是,监控在硬件系统1中运行的数据流,因为对此需要特定的硬件。然而,借助于特定的硬件的故障分析对于未改变的实际硬件来说仅受限地有说服力。

[0070] 按照本发明,为了获得对本身不可见的数据流的认知,将硬件系统1映射为在仿真环境4中能运行的仿真模型5。

[0071] 仿真环境4可以是任意的、适合的仿真平台,所述仿真平台能够在硬件系统1的仿真模型5中对在执行数据处理任务期间出现的数据流进行仿真,以便使所述数据流可观察。

[0072] 在仿真环境4中的仿真允许观察和记录由各个硬件组件7执行的数据流。优选地,将所述观察和记录限制到关键数据上,所述关键数据在执行仿真之前经由相应的用户界面25规定控制和监控系统24并且在仿真环境4的“定义关键数据”6中提供。控制和监控系统24允许用户经由用户界面25创建所有定义和指令,所述定义和指令对于在仿真环境4中的仿真的流程是必需的,包括仿真模型5的建模、关键数据6的定义、仿真运行的开始和中止以及参数数据13或测试用例的定义或选择。

[0073] 允许将硬件系统1映射到仿真模型5中的技术和系统是本技术领域已知的,并且已经例如已经由Ye,X.,Fan,D.,Sun,N.,Tang,S.,Zhang,M.,&Zhang,H.在文章“SimICT:A fast and flexible framework for performance and power evaluation of large-scale architecture”(2013年9月,发表在:2013年低功耗电子与设计国际研讨会论文集(Proceedings of the 2013 International Symposium on Low Power Electronics and Design)(273-278页),IEEE出版社)中描述。

[0074] 仿真模型5例如也可以仅映射较大的整体系统的一部分、例如如果并非整体系统的所有组成部分在软件中可仿真的话,如果创建完整的仿真模型太耗费的话或者如果这样的仿真需要太多的计算时间的话。在这种情况下,与数据流不相关的部分可以作为真实的硬件元件耦合到仿真环境4上、例如以硬件在环(Hardware-in-the-Loop)仿真的方式。以类似的方式,在其上运行仿真模型5的仿真环境4可以与一个(或多个)另外的仿真环境交换数据,其中,另外的仿真环境对数据流不必按照本发明的方法来检查的元件进行仿真,并且因此能够较简单地设计。

[0075] 在仿真中,仿真模型5执行从应用软件12得出的数据处理任务2,其中,基于预先定义的测试用例确定参数数据13。在此,例如可以将软件程序作为整体或所述整体的各个部分来处理。在此,所述测试用例例如可以按照硬件系统1的尽可能真实的流程来设计,但是所述测试用例也可以表示特别的场景或者涉及在界限范围内的运行条件。用于创建和优化与用于车辆的行驶辅助系统相关联的测试用例的方法例如在WO 2015/067649 A1中公开并且在此称为“测试场景”。在仿真环境4使得仿真模型5按照这样的测试用例运行期间,创建、传输、处理/链接和删除在仿真模型5中的关键数据的物理拷贝由控制和监控系统24观察和记录。

[0076] 在仿真结束之后(或在此期间),根据所记录的数据由控制和监控系统24创建分析结果14,所述分析结果必要时也包括图形表示。在此,所述分析结果14尤其是示出经由有线

的和/或无线的数据连接的关键数据的传输过程,以及在硬件系统1的建模为仿真模型5的表示中的任意存储位置处的关键数据的物理拷贝的创建和删除。特别感兴趣的是关键数据的实例,所述实例在仿真结束之后可以存在于存储器单元3中。这可以是如下情况,即,关键数据6的拷贝在仿真结束之后(或者在限定的数据处理任务2结束之后或者在删除在原始存储位置处的关键数据6之后)还未被删除或者覆盖,并且因此以未公开的方式在所述存储器单元3之一中的确定的地址处还存在。

[0077] 以有利的方式,在分析结果14中可以将存储器单元3和/或通信装置8和/或通信单元9分别划分到“允许域”15和“禁止域”16中。所述域例如可以根据整个存储器单元3、或者在存储器单元3中的确定的地址空间、根据总线地址或者根据硬件组件7来定义,其中,所述定义或划分同样可以经由控制和监控系统24来进行。

[0078] 仿真模型5可以按照需要以可定义的细节程度来表示要执行的硬件系统1。例如,所述仿真模型5可以在CPU命令集(CPU指令层级)的层级上、或者在硬件门层级上建模。所需要的细节程度与要测试的攻击类型有关。为了发现由于软件错误的非期望的数据流,例如在CPU的指令层级上的仿真就已经足够。为了发现由于硬件漏洞(例如在作为SPECTRE和MELTDOWN已知的安全漏洞的情况下)或缓存攻击的非期望的数据流,必须在硬件层级/门层级上进行仿真。

[0079] 尤其是,所述硬件系统1也可以包括子系统、例如这样的具有存储器直接访问、外围设备等的子系统以及通信接口。

[0080] 图2示出硬件系统1的具体示例,在所述硬件系统上运行操作系统11,利用所述操作系统可以执行应用软件12。硬件系统1按照常见的计算机架构来构造,其中,处理器17包括处理器核心18、处理器缓存19和处理器IO20。处理器17可以经由总线架构21与另外的组件22通信。包括处理器核心18、处理器缓存19、处理器IO20、总线架构21和另外的组件22的单元可以看待为网络物理系统,所述网络物理系统与(真实的)环境(典型地经由传感器和致动器)交换。这样的网络物理系统例如可以是汽车控制器(ECU)。

[0081] 在硬件系统1上运行有操作系统11,利用所述操作系统11可以执行应用软件12。

[0082] 按照与图1进行的一般性描述相关联的意义,图2的处理器17例如对应于第一硬件组件7a,处理器核心18对应于处理器单元10,处理器缓存19对应于存储器单元3,处理器IO20对应于通信单元9,总线系统21对应于通信装置8,并且另外的组件22可以包括另外的硬件组件7。如已经提到的那样,另外的建模、例如在较详细的层级上的建模也是可能的。

[0083] 在图3中在仿真模型5中示例性地示出图2的硬件系统1的建模,所述仿真模型可以由仿真环境4在考虑的操作系统11和应用软件12的情况下作为对硬件系统1的仿真来运行。

[0084] 图4以示意图示出了在对应于按照本发明的方法的测试流程中执行的步骤。测试员23可以经由控制和监控系统24的用户界面25操作仿真环境4,其中,在图4的图示中已经完成对仿真模型5的建模。在定义步骤I中,测试员定义关键数据6,其中,所述定义尤其是基于对硬件系统1和在所述硬件系统上运行的应用软件12的分析。待监控的关键数据6由用户例如通过特定的标记来确定,其中,这可以直接在软件中进行、亦即在编译之前进行。为此,至少部分地需要访问应用软件12的源代码和/或调试信息、例如符号文件。必要时,确定关键数据的定义也可以通过指明文件属性、例如对于数据内容、对于形成时刻、对于处理时刻、对于在存储器中的地址等的属性来进行,这在很大程度上在不访问源代码的情况下也

是可能的。

[0085] 因此,与本公开的技术方案相关联地,“关键数据”尤其是表示如下数据,所述数据在定义步骤中或是由待执行的人员(例如程序员或测试员23)或是根据预先定义的特性定义为关键数据。用于定义关键数据的标准尤其是与相应的个别情况和待检查的硬件系统和/或数据处理任务相关。本领域技术人员在认知在此公开的教导的情况下能够进行对关键数据的定义。

[0086] 与定义关键数据6相关联地,可以建立规则、例如描述“界限”,所监控的关键数据6不允许“超过”所述界限。所述界限可能涉及硬件的确定的部分、确定的存储器区域、在CPU中的确定的部分等,并且分别将所述资源划分到允许域15和禁止域16中。所述定义也可以包含另外的限制、例如确定的时间段,或者与系统状态的相关性。作为用于定义所述域的另外的特征,必要时可以使用如下情境,在所述情境中执行所述过程、例如哪个(或哪些)用户执行所述过程、或者处理器是否以用户模式或核心模式运行等。

[0087] 在定义步骤I期间也进行测试用例的选择或定义,所述测试用例应该在仿真的过程中处理。也可以在所述步骤中进行仿真模型5的建模,或者匹配所述建模。

[0088] 应当指出的是,仿真模型5的建模、关键数据6的定义、测试用例的选择以及规则和界限的定义的步骤可以以任意的次序或者也可以彼此并行地或在迭代步骤中执行。

[0089] 在步骤II中,待检查的应用软件12在仿真环境中根据应用情形或测试情形设置,并且在步骤III中,所述仿真在仿真环境4中启动并且由控制和监控系统24监控运行中的仿真。因为待测试的应用软件和必要时还有在所述应用软件之下的待测试的部分(框架、运行时间库、操作系统…)以及硬件在仿真环境4中“运行”,对关键数据的所有访问、或者关键数据(及其拷贝)的流可以观察、监控并且与所定义的规则比较。违反规则可以被记录和/或作为警告报告,并且因此形成对可能的数据漏洞的发现。此外,通过记录对于所述发现的情境和历史(在仿真环境4中的完整流程可以通过踪迹方法来记录)可以在进一步地诊断(或调试)中提供重要支持。

[0090] 控制和监控系统24至少部分地访问所述仿真环境4的内部寄存器和状态,这类似于典型的调试器接口。

[0091] 因此,关键数据6、对所述关键数据的每次访问以及通过软件和硬件的活动可以从所述数据中导出的所有变化(例如在CPU寄存器中的标志)被观察和监控。所述关键数据的路径可以由控制和监控系统24存储并且在分析结果14中例如以轨迹(“踪迹”)的形式以树形结构表示(步骤IV)。

[0092] 在图5中示例性地示出了用于具有树形结构的关键数据6的这样的分析结果14。x轴表示仿真的时间经过,在y轴上示出硬件的划分到允许域15和禁止域16中的部段。

[0093] 在图5中示出的示例中,在时刻(a)开始监控关键数据6(例如当关键数据载入到硬件组件7的存储器中时)。在时刻(b),在过程步骤的过程中创建关键数据的拷贝,所述拷贝称为第一实例6’。所述第一实例位于与允许域15相配的存储器区域中。在时刻(c),从第一实例中通过拷贝产生第二实例6”并且将其存储在与禁止域16相配的存储器中。所述过程由控制和监控系统24识别和报告。必要时,可以产生警告通知和/或中止仿真运行。在时刻(d),关键数据的原始数据被破坏(亦即、例如被覆盖或删除)。然而,所述第一实例6’在此时刻仍存在(然而在允许域中)。这也被标记和报告。按照这种方式,可以识别所监控的关键数

据6的所有实例以及对其的访问。典型地,新的实例形成新的子树的开始节点。

[0094] 在实践中,上面描述的分析结果14例如可以记录以下过程:所监控的数据从外部存储器载入到CPU寄存器中。但是典型地,因此可以在缓存中(可能甚至在多个级别中)产生拷贝,即使在CPU寄存器中删除或改写数据,所述拷贝也存在。因此,所述拷贝(亦即、所述第一实例6')必须被重新监控,并且因此在图形中在相应的位置中形成(例如与缓存相关联的)新分支。经由总线传输数据也能实现使潜在的连接在相同总线上的所有接收方在此时刻读取(并且接着继续拷贝或处理)所述数据。这也利用新的分支在图形中记录和继续跟踪。

[0095] 另外的可能性是处理所监控的数据。例如通过移位(Shift)命令将数据二进制地移位多个位元位置并且因此改变数据。结果与所监控的数据相关并且因此同样作为新的分支在图形中记录和继续跟踪。

[0096] 在此公开的方法例如能实现对如下工具的新家族的开发,所述工具用于测试和用于诊断特性、例如尤其是在车辆中的控制器和总线系统中的"保密性"和"私有性"。尤其是与开发自动辅助的和自主的车辆相关地,用于动力传动系、行驶功能、车辆通信、乘客舒适度(也称为eHealth)和有效负载管理的将来的控制器越来越多地处理与个人相关的或对任务关键的数据。这样的数据仅允许以定义的方式和方法公开或转移,并且绝不能随机地或由于针对性地攻击而被截取。借助于本发明能实现,对不同的硬件系统以任意的细节深度测试与安全相关的数据流。所述测试也可缩放到任意的范围上,其中,例如可以仿真多个交互的系统。通过同样可缩放的细节深度,错误可以在不同的层级上、这可能在应用软件中、在操作系统中或已经在管理程序中、或者更深地直接在硬件实现中(这例如可以与旁道攻击相关)跟踪。

[0097] 附图标记列表

[0098]	硬件系统	1
[0099]	数据处理任务	2
[0100]	存储器单元	3
[0101]	仿真环境	4
[0102]	仿真模型	5
[0103]	关键数据	6
[0104]	硬件组件	7
[0105]	通信装置	8
[0106]	通信单元	9
[0107]	处理器单元	10
[0108]	操作系统	11
[0109]	应用软件	12
[0110]	参数数据	13
[0111]	分析结果	14
[0112]	允许域	15
[0113]	禁止域	16
[0114]	处理器	17

[0115]	处理器核心	18
[0116]	处理器缓存	19
[0117]	处理器IO	20
[0118]	总线系统	21
[0119]	另外的组件	22
[0120]	测试员	23
[0121]	控制和监控系统	24
[0122]	用户界面	25

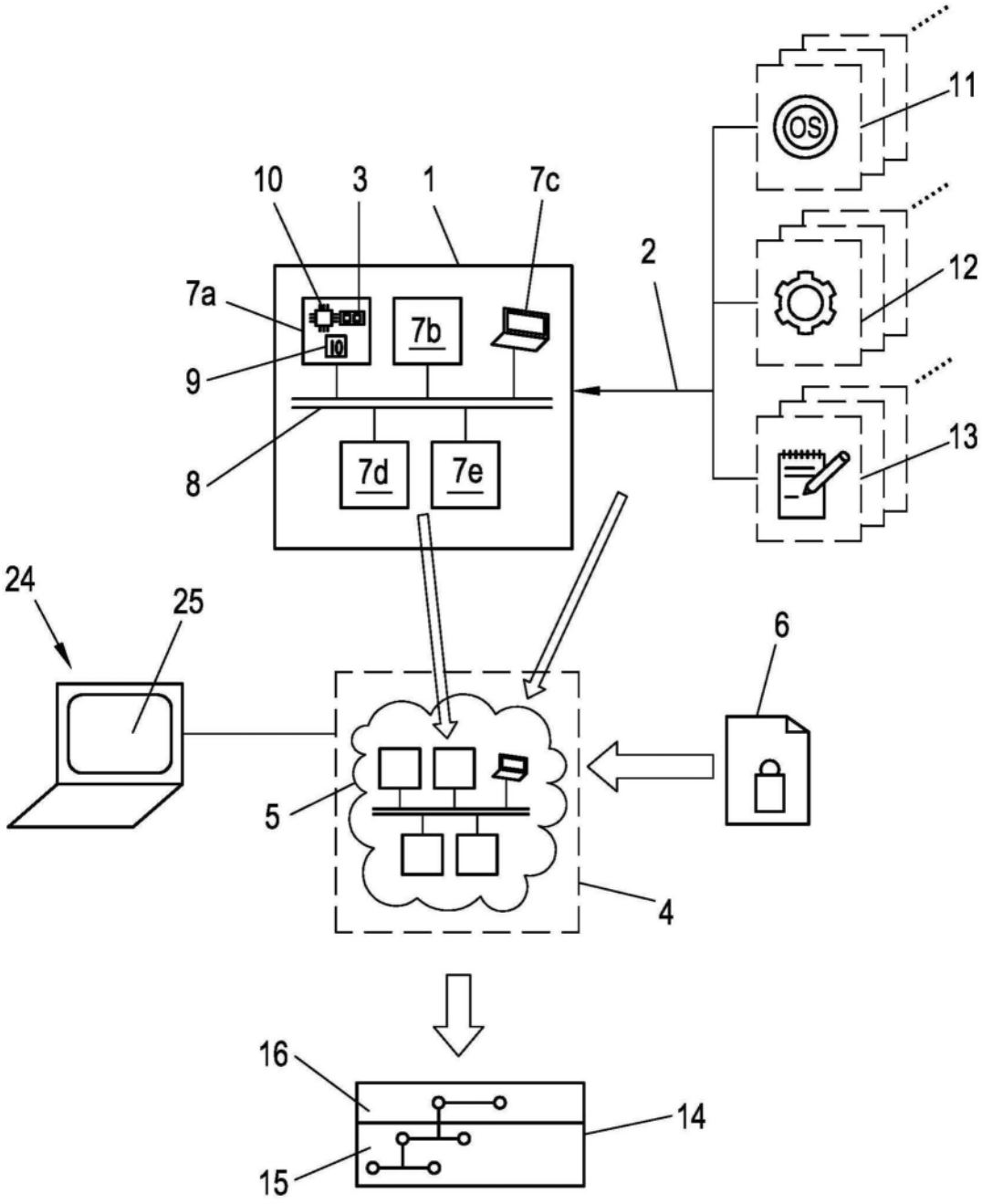


图1

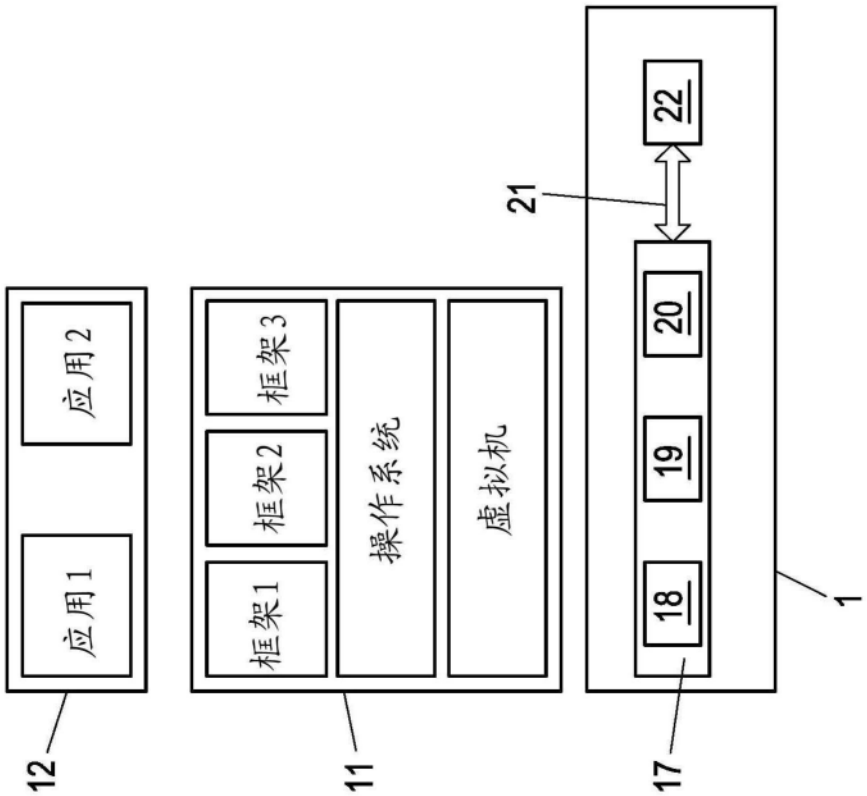


图2

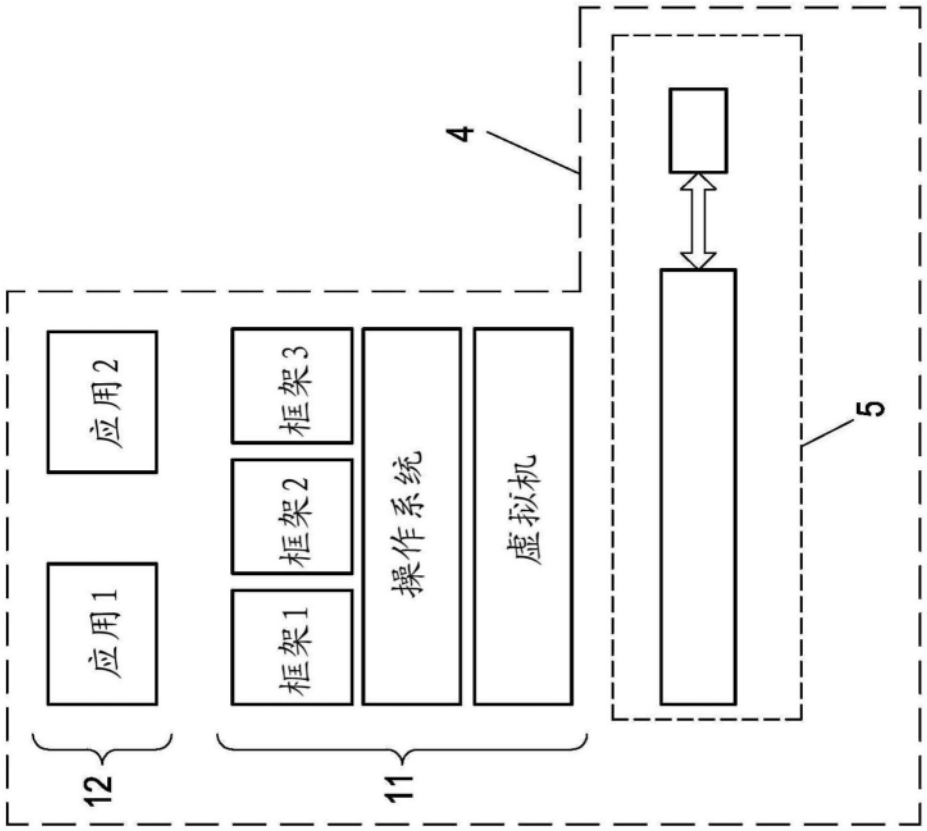


图3

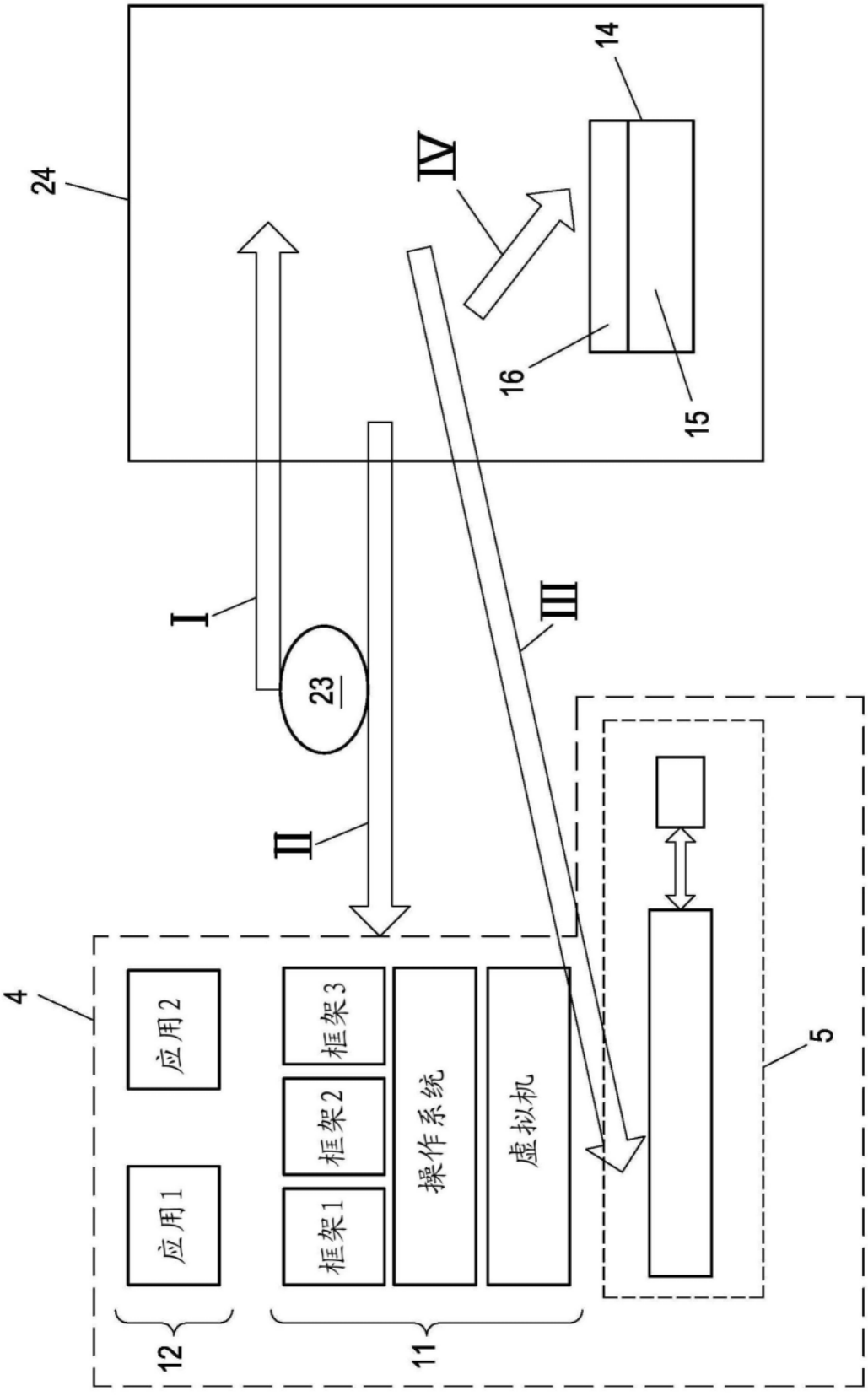


图4

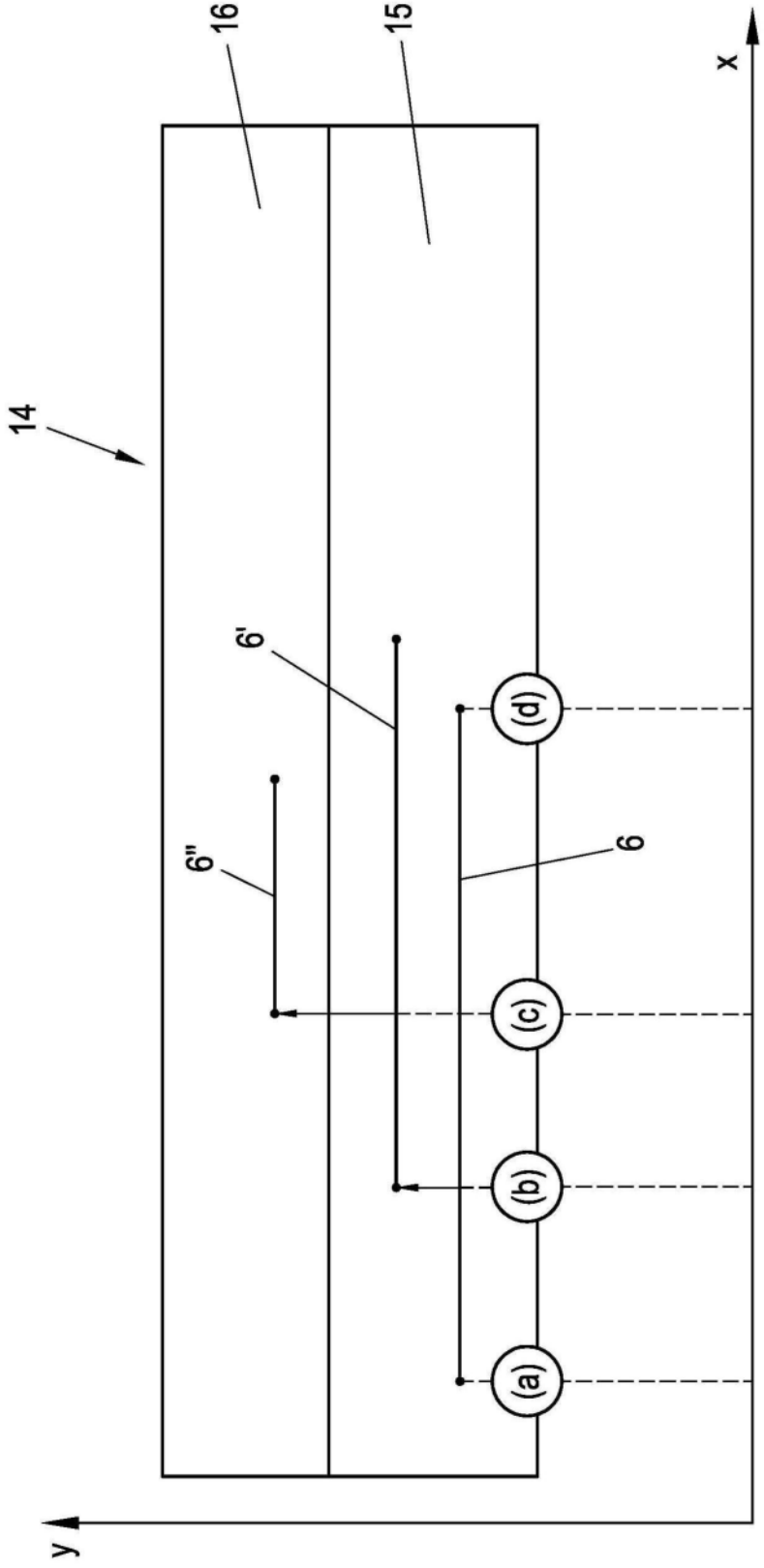


图5