



(12)发明专利

(10)授权公告号 CN 103516561 B

(45)授权公告日 2017.11.28

(21)申请号 201310420124.0

(56)对比文件

(22)申请日 2013.09.13

US 7016807 B2, 2006.03.21,

(65)同一申请的已公布的文献号

CN 102722438 A, 2012.10.10,

申请公布号 CN 103516561 A

CN 101681280 A, 2010.03.24,

(43)申请公布日 2014.01.15

审查员 黄欣欣

(73)专利权人 汉柏科技有限公司

地址 300384 天津市西青区华苑产业区海  
泰西18号西3楼104室

(72)发明人 李鹏

(74)专利代理机构 北京中政联科专利代理事务  
所(普通合伙) 11489

代理人 陈超

(51)Int.Cl.

H04L 12/26(2006.01)

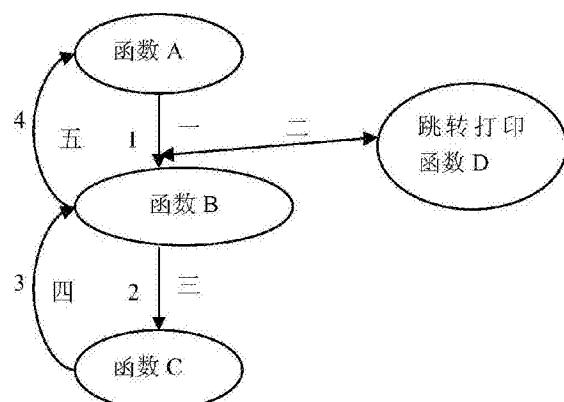
权利要求书1页 说明书4页 附图3页

(54)发明名称

一种网络系统的调试方法

(57)摘要

本发明提供一种网络系统的调试方法,该方法包括:确定要调试的函数,下发跳转打印指令;获取所述被调试函数的名称和执行地址;根据所述被调试函数的名称和执行地址,用预制的跳转打印函数修改被调试函数的栈,使得跳转打印函数先于被调试函数执行;执行跳转打印函数,打印调试信息。该方法只要预先做好一条或者多条打印信息,当网络设备出现问题时,通过一个动态输入命令来使原本没有打印信息的函数自动打印预设的信息,以便开发人员调试定位信息。这种方法具有较广泛的应用场景,不仅仅局限于已经发布的产品,对于正处在开发阶段的产品来说,同样可以使用该方法调试定位问题。



1. 一种网络系统的调试方法,其特征在于,该方法包括:

通过CLI或者WEB下发要调试的函数名称;

获取所述被调试函数的名称和执行地址;

根据所述被调试函数的名称和执行地址,用预制的跳转指令修改被调试函数的栈,使得跳转打印函数先于被调试函数执行;

执行程序,打印调试信息;

找出打印出函数名的函数与未打印出函数名的函数;

判断相近的打印出的函数名与未打印出的函数名之间是否还包含有其他函数;如果没有其他函数,则判断该未打印出函数名的函数就是出错的函数;

如果还有其他函数,则继续返回到确定要调试的一个或多个函数,继续顺序执行,直至相邻的打印出的函数名与未打印出的函数名之间没有其他函数,则判断该未打印出函数名的函数就是出错的函数。

2. 根据权利要求1所述的网络系统的调试方法,其特征在于,在所述通过CLI或者WEB下发要调试的函数名称之前还包括:预先制作跳转指令的二进制码,所述跳转指令的二进制码包括跳转到打印函数的地址。

3. 根据权利要求1所述的网络系统的调试方法,其特征在于,执行跳转打印函数时包括下列步骤:

保存被调试函数的入参和返回地址;

打印调试信息;

恢复被调试函数的栈信息;

修改跳转打印函数的栈信息。

4. 根据权利要求1所述的网络系统的调试方法,其特征在于,修改被调试函数的栈包括:保存被调试函数二进制代码的前几个字节,并修改为预先写好的跳转指令的二进制码,使得第一个指令跳转到跳转打印函数。

5. 根据权利要求3所述的网络系统的调试方法,其特征在于,所述的修改跳转打印函数的栈信息是修改跳转打印函数的后面几字节的二进制代码,使得跳转打印函数的栈得到正常回收并返回到被调试函数。

## 一种网络系统的调试方法

### 技术领域

[0001] 本发明属于网络设备的调试领域,特别涉及一种通过动态插入函数的方式对网络系统进行调试的方法。

### 背景技术

[0002] 当今是互联网快速发展的时代,各种各样的网络产品层出不穷,用户对网络产品的需求也逐渐增多,各个厂商也不断的给用户更加丰富的功能和体验,成为网络设备上网络通信过程中必不可少的环节。随着近些年的快速发展,网络设备也越来越多,越来越复杂,交换机、路由器产品也仅仅是这些网络产品中一小部分,网络的部署也越来越复杂,从单机设备部署,框架式设备部署,到现在发展为数据中心、云部署。由于设备功能、网络部署都变得越来越多,越来越复杂,因此一种良好的调试手段就显得尤为重要。在产品发布之前的开发阶段通常会有各种各样的调试手段,但是在产品发布后,通常的调试手段都比较少,难以处理较复杂的场景。

[0003] 现有技术通常是在代码中添加大量打印或者日志信息,当网络设备出现问题时通过后台查看日志信息,或者用户终端的打印信息来分析可能出现的问题。但是这就需要提前将打印信息做好,如果打印信息没有添加到发布的版本中,那么就无法获取相应信息。

[0004] 同时,当设备出现问题时,仅仅看打印信息是不够的,研发人员通常需要一种方法获知某个代码流程是否走到了,或者更细微的某个函数是否得到了执行,但是遗憾的是,当前要不没有办法要不还要借助第三方工具。当发布的网络设备出现问题时,研发人员通常无法通过跟踪源代码函数来确定某个函数是否得到执行。

### 发明内容

[0005] 本发明鉴于上述问题而提出,其目的是提供一种网络系统的调试方法,实现动态打印,只要预先做好一条或者多条打印信息,当网络设备出现问题时,通过一个动态输入命令来使原本没有打印信息的函数自动打印预设的信息,以便开发人员调试定位信息。

[0006] 为达到上述目的,本发明的技术方案是:一种网络系统的调试方法,该方法包括:

[0007] 通过CLI或者WEB下发要调试的函数名称;

[0008] 获取所述被调试函数的名称和执行地址;

[0009] 根据所述被调试函数的名称和执行地址,用预制的跳转打印函数修改被调试函数的栈,使得跳转打印函数先于被调试函数执行;

[0010] 执行跳转打印函数,打印调试信息。

[0011] 进一步的,在所述确定要调试的被调试函数,跳转到跳转打印函数之前还包括:预先制作跳转指令的二进制码,所述二进制码包括跳转打印函数的地址。

[0012] 进一步的,执行跳转打印函数包括:

[0013] 保存被调试函数的入参和返回地址;

[0014] 打印调试信息;

- [0015] 恢复被调试函数的栈信息；  
[0016] 修改跳转打印函数的栈信息。  
[0017] 进一步的，修改被调试函数的栈包括：保存被调试函数二进制代码的前几个字节，修改为预先写好的跳转指令的二进制码，使得第一个指令跳转到跳转打印函数。  
[0018] 进一步的，所述的修改跳转打印函数的信息是修改跳转打印函数的后面几字节的二进制代码，使得跳转打印函数的栈得到正常的回收并返回到被调试函数。  
[0019] 本发明方法的有益效果是：使用本专利描述的方法，当想查看某函数是否得到执行时，只需下发命令就可以了，如果确实得到了执行，会有对应的函数名称打印出来，如果没有得到执行，那么就没有打印信息。  
[0020] 该方法使用简单，且具有通用性，减少了各种调试打印信息的开发量，以及维护工作量，大大提高了工作效率。  
[0021] 这种方法具有较广泛的应用场景，不仅仅局限于已经发布的产品，对于正处在开发阶段的产品来说，同样可以使用该方法调试定位问题。

## 附图说明

- [0022] 图1是本发明的网络系统的调试方法的原理示意图；  
[0023] 图2是本发明的网络系统的调试方法的执行流程图；  
[0024] 图3是本发明的网络系统的调试方法的使用流程图。

## 具体实施方式

[0025] 为使本发明的目的、技术方案和优点更加清楚明了，下面结合具体实施方式并参照附图，对本发明进一步详细说明。应该理解，这些描述只是示例性的，而并非要限制本发明的范围。此外，在以下说明中，省略了对公知结构和技术的部份描述，以避免不必要的混淆本发明的概念。

- [0026] 图1显示了本发明的网络系统的调试方法的原理示意图。  
[0027] 图中示例性的给出了最简单的函数调用关系，其中函数B是分析确定的要调试的函数，函数D是按本方法制作的跳转打印函数。  
[0028] 如图1所示，正常的调试方法中的函数调用过程是：函数A调用函数B，函数B调用函数C，函数C执行完后返回到函数B，函数B执行完后返回到函数A，而跳转打印函数D根本没有被执行到。上述执行顺序见图1的数字序号1-2-3-4。  
[0029] 本发明的调试方法中，函数的执行顺序被打断了，变成函数A调用函数B，而此时函数B没有得到执行反而去执行跳转打印函数D，而后跳转打印函数D执行完后返回到函数B，此时函数B才开始得到执行，然后调用函数C，而后就是顺序执行了。上述执行顺序见图1中文序号一-二-三-四-五。  
[0030] 图2显示了本发明网络系统的调试方法的执行流程图。  
[0031] 如图2所示，本发明网络系统的调试方法的执行包括以下步骤：  
[0032] 首先是准备工作(步骤301)，包括预制跳转指令的二进制机器码和制作跳转打印函数D，制作跳转二进制码的作用是跳转到跳转打印函数D的地址；  
[0033] 接下来就是，确定需要调试的函数B(步骤302)，通过CLI或者WEB下发命令(步骤

303), 获取函数B的函数名称和地址(步骤304), 分析需要打断执行的函数B的执行地址, 并将函数B的名称附给一个全局变量。

[0034] 接下来就是跳转到跳转打印函数D(步骤305);

[0035] 当跳转打印函数D开始执行时, 完成的步骤包括:

[0036] (1) 保存函数B的信息(步骤306), 包括函数B的入参和返回地址。保存函数B的入参, 对于x86平台来说就是EBP(寄存器)以及其偏移对应地址中的值, 对于Mips(Microprocessor without interlocked piped stages)平台对应a0,a1,a2,a3寄存器。保存这些值到跳转打印函数D的栈空间。需要说明的是Mips是一种CPU架构的名字,Mips和x86,arm都是对应的CPU架构。

[0037] 保存函数B的返回地址。这样程序才能最终返回到函数A中。

[0038] (2) 打印函数B的调试信息(步骤307)。这里以打印函数名举例, 由于前面在全局变量中保存了函数B的名字, 因此这里就可以打印出该函数的名字了。

[0039] 当然, 如果函数B得到了执行则会打印出调试信息(步骤311), 如果函数B没有执行则无打印输出(步骤312)。

[0040] (3) 恢复函数B的信息(步骤308)。恢复之前保存在跳转打印函数D栈空间的函数B的入参、返回地址等信息。

[0041] 这样函数B就正常了, 但是由于跳转打印函数D还没有返回, 因此函数B还得不到执行。

[0042] (4) 所以要修改跳转打印函数D的信息(步骤309)。

[0043] 需要说明的是, 大部分函数开始时会进行开栈, 在函数结束时会收栈。进入跳转打印函数D后也会有这些操作, 如果在跳转打印函数D执行完毕后直接跳转到函数B, 那么实际上跳转打印函数D的栈没有回收, 会造成不可预知的问题。因此需要将跳转打印函数D的最后面的二进制数值改掉, 这样跳转打印函数D会自然地返回到函数B, 同时也做了栈的回收工作。

[0044] 跳转打印函数D执行完了, 就会打印出函数B的名称。

[0045] 现在处于函数B中, 且函数B的栈信息、入参、返回地址等信息都得到了恢复, 就好像从来没有变化过一样, 因此函数B可以继续顺序执行(步骤310)。

[0046] 如果函数B得到执行了, 则会打印出函数B的名称, 说明程序出错的地方在函数B的后面。

[0047] 当然, 如果程序在函数B的前面就已经出错了, 这时函数B是不会执行的, 因此也就没有打印信息。所以如果没有打印到函数的名称, 就说明出错的地方在函数B前面。

[0048] 如上所述, 根据本发明的网络系统的调试方法, 可以用于判断程序出错的位置。

[0049] 图3示出了本发明网络系统的调试方法的使用流程图。

[0050] 当程序中的调用函数很多时, 查找出错函数可以采用二分法查找。

[0051] 首先预制跳转打印函数及跳转指令(步骤401);

[0052] 然后确定要调试的一个或者多个函数(步骤402);

[0053] 其次就是通过CLI或者WEB下发跳转打印命令(步骤403);

[0054] 执行程序, 查看打印的信息(步骤404);

[0055] 找出打印出函数名的函数与未打印出函数名的函数(步骤405);

[0056] 然后判断相近的打印出的函数名与未打印出的函数名之间是否还包含有其他函数(步骤406)；

[0057] 如果没有其他函数了则该未打印出函数名的函数就是出错的函数(步骤407)。

[0058] 如果还有其他函数，则继续返回到确定要调试的一个或多个函数(步骤402)，继续顺序执行，直至相邻的打印出的函数名与未打印出的函数名之间没有其他函数，则该未打印出函数名的函数就是出错的函数。

[0059] 公知的方法是，如果要调试函数A，比如仅仅是打印函数A的名字，那么需要在代码里面写好这句话，无论什么时候，只要函数A得到执行，那么就会打印函数A的名字。如果要调试函数B，也需要上述过程。如果要调试的函数很多，那么就要做很多上述工作，操作非常不方便。

[0060] 但是如果应用本发明的调试方法，只需要预先制作一个跳转打印函数，以及跳转指令的二进制码就可以了。无论你要调试哪个函数，都可以使用这个方法，而且打印信息只生效一次，不会一直存在，这样不仅使用方便，同时也不会带来过多的打印输出。

[0061] 应当理解的是，本发明的上述具体实施方式仅仅用于示例性说明或解释本发明的原理，而不构成对本发明的限制。因此，在不偏离本发明的精神和范围的情况下所做的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。此外，本发明所附权利要求旨在涵盖落入所附权利要求范围和边界、或者这种范围和边界的等同形式内的全部变化和修改例。

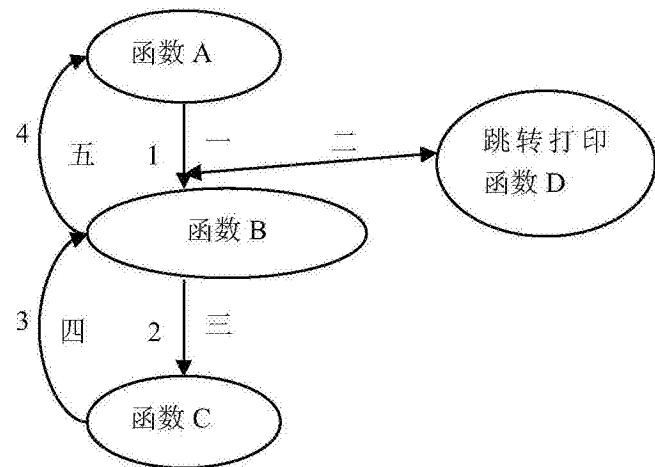


图1

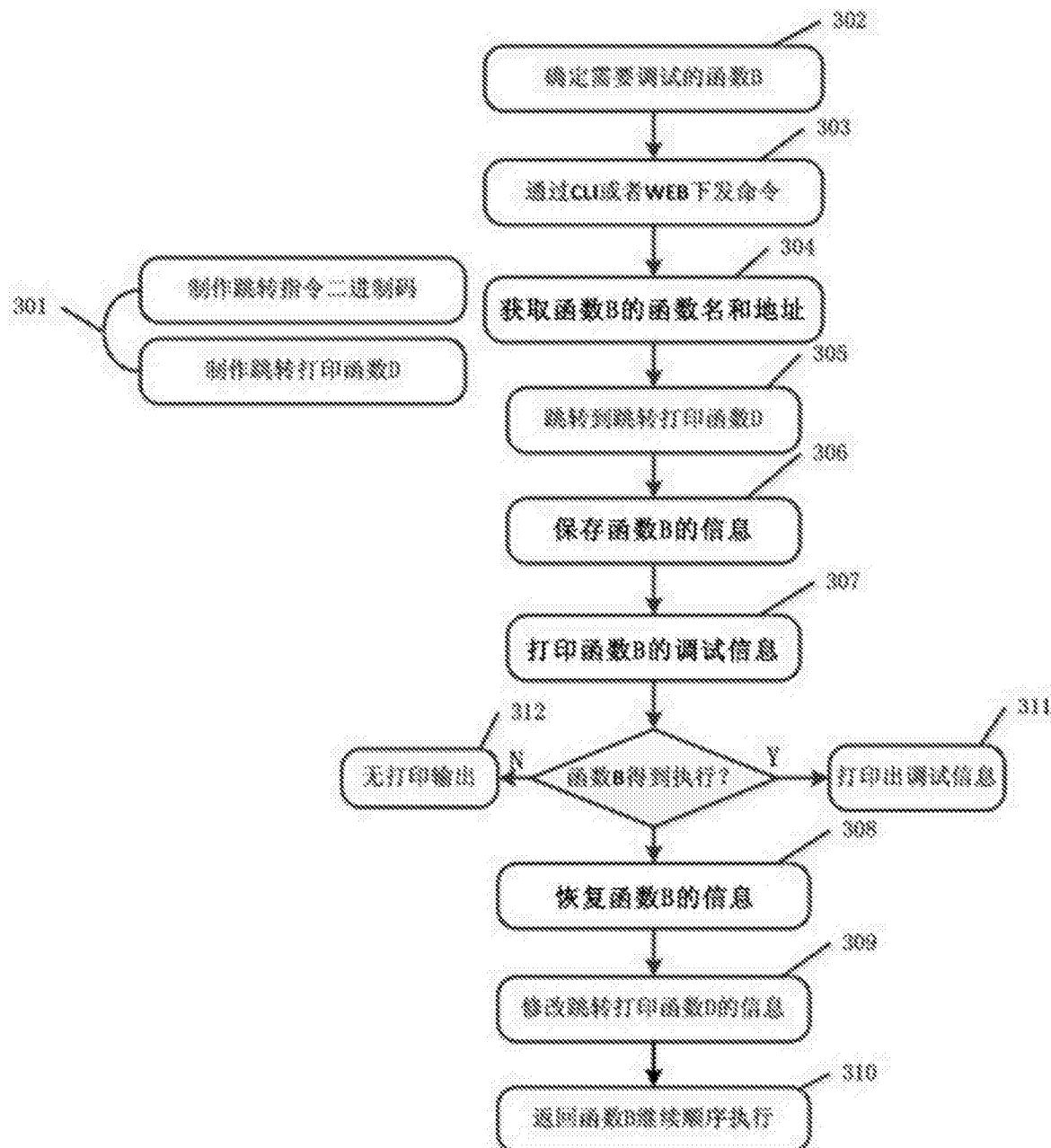


图2

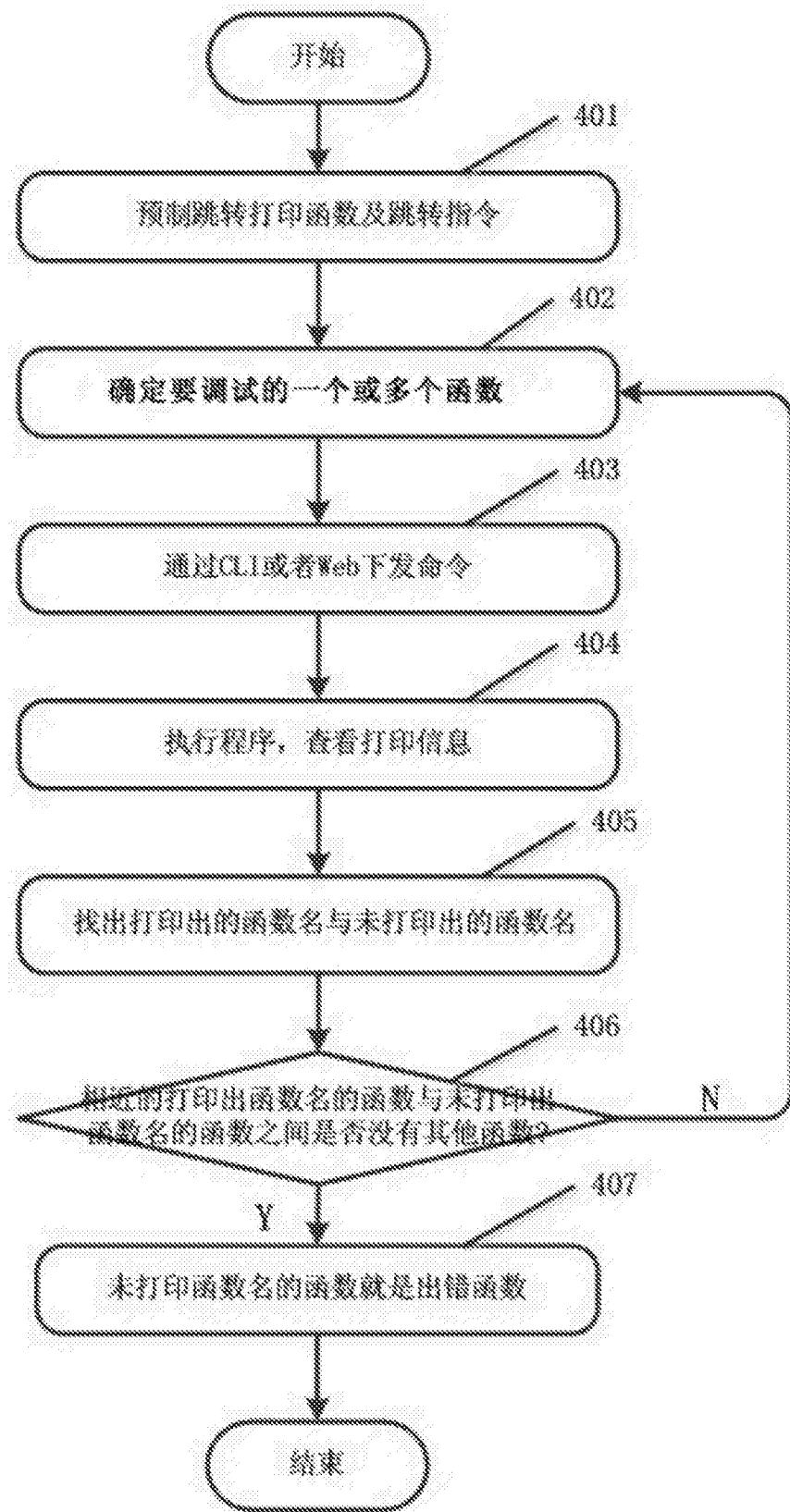


图3