

(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国际局

(43) 国际公布日
2022 年 9 月 29 日 (29.09.2022)



(10) 国际公布号
WO 2022/199331 A1

(51) 国际专利分类号:
G06F 11/07 (2006.01)

(21) 国际申请号: PCT/CN2022/078365

(22) 国际申请日: 2022 年 2 月 28 日 (28.02.2022)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
202110303887.1 2021年3月22日 (22.03.2021) CN

(71) 申请人: 北京字节跳动网络技术有限公司 (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD.) [CN/CN]; 中国北京市石景山区实兴大街30号院3号楼2层B-0035房间, Beijing 100041 (CN)。

(72) 发明人: 刘洪凯 (LIU, Hongkai); 中国北京市海淀区知春路63号中国卫星通信大厦今日头条小邮局, Beijing 100086 (CN)。

(74) 代理人: 北京知帆远景知识产权代理有限公司 (ZHIFAN & PARTNERS); 中国北京市海淀区阜成路73号裕惠大厦B座805, Beijing 100142 (CN)。

(81) 指定国(除另有指明, 要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK,

(54) Title: METHOD AND APPARATUS FOR OBTAINING INFORMATION OF STACK FRAME IN CALL STACK, DEVICE, AND MEDIUM

(54) 发明名称: 用于获取调用栈中栈帧信息的方法、装置、设备和介质

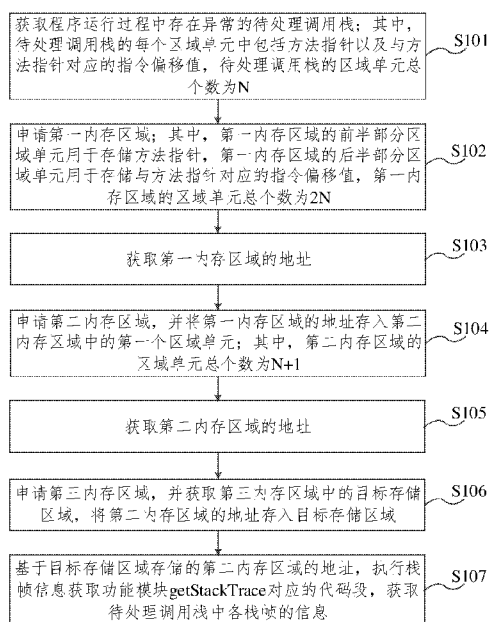


图 1

(57) Abstract: Embodiments of the present disclosure relate to a method and apparatus for obtaining information of a stack frame in a call stack, a device, and a medium. The method comprises: obtaining a call stack to be processed that is abnormal in a program running process, each area unit of said call stack comprising a method pointer and an instruction offset value corresponding to the method pointer; applying for a first memory area, the first memory area being used for storing the method pointer and the corresponding instruction offset value; applying for a second memory area; storing an address of the first memory area into a first area unit in the

SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG,
US, UZ, VC, VN, WS, ZA, ZM, ZW。

- (84) 指定国(除另有指明, 要求每一种可提供的地区保护): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:

— 包括国际检索报告(条约第21条(3))。

second memory area; applying for a third memory area and storing an address of the second memory area into a target storage area in the third memory area; and on the basis of the address of the second memory area stored in the target storage area, executing a code segment corresponding to a stack frame information obtaining functional module getStackTrace to obtain information of each stack frame in said call stack. According to the embodiments of the present disclosure, detailed information of the stack frame in said call stack is stably obtained, and the compatibility is high.

(57) 摘要: 本公开实施例涉及一种用于获取调用栈中栈帧信息的方法、装置、设备和介质, 该方法包括: 获取程序运行过程中存在异常的待处理调用栈; 待处理调用栈的每个区域单元中包括方法指针以及与方法指针对应的指令偏移值; 申请第一内存区域, 用于存储方法指针以及对应的指令偏移值; 申请第二内存区域; 将第一内存区域的地址存入第二内存区域中的第一个区域单元; 申请第三内存区域, 并将第二内存区域的地址存入第三内存区域中的目标存储区域; 基于目标存储区域存储的第二内存区域的地址, 执行栈帧信息获取功能模块getStackTrace对应的代码段, 获取待处理调用栈中各栈帧的信息。本公开实施例实现了稳定地获取调用栈中栈帧的详细信息, 且兼容性高。

用于获取调用栈中栈帧信息的方法、装置、设备和介质

相关申请的交叉引用

本申请要求于 2021 年 03 月 22 日提交的，申请号为 202110303887.1、发明名称为“用于获取调用栈中栈帧信息的方法、装置、设备和介质”的中国专利申请的优先权，该申请的全部内容通过引用结合在本申请中。

技术领域

本公开涉及计算机技术领域，尤其涉及一种用于获取调用栈中栈帧信息的方法、装置、设备和介质。

背景技术

在应用程序运行过程中，通过栈回溯，可以得到每个栈帧包括的方法指针（例如 ArtMethod 的指针）和指令偏移值（DexPc），但是不能直接得到描述当前方法的详细信息，例如方法名和源码行号等信息。

为了得到描述方法的详细信息，在现有方案中，一方面，可以将方法（例如 ArtMethod）指针作为参数调用 PrettyMethod 函数得到其相应方法的字符串描述信息；另一方面，可以通过一些底层技术得到方法对应的 dex_file 信息，然后基于 dex_file 信息获取方法所在类名、方法名等信息。

然而，前述第一种方法中，只能得到方法的名字信息，无法得到其相应的行号信息，进而不能快速定位源码位置；前述第二种方法中，由于需求采用较多底层技术，实现复杂，稳定性难以保证，并且针对不同系统版本的兼容性也存在不确定因素，可能需要涉及较大的适配工作量。

发明内容

为了解决上述技术问题或者至少部分地解决上述技术问题，本公开实施例提供了一种用于获取调用栈中栈帧信息的方法、装置、设备和介质。

第一方面，本公开实施例提供了一种用于获取调用栈中栈帧信息的方法，包括：

获取程序运行过程中存在异常的待处理调用栈；其中，所述待处理调用栈的每个区域单元中包括方法指针以及与所述方法指针对应的指令偏移值，所述待处理调用栈的区域单元总个数为 N；

申请第一内存区域；其中，所述第一内存区域的前半部分区域单元用于存储所述方法

指针, 所述第一内存区域的后半部分区域单元用于存储与所述方法指针对应的指令偏移值, 所述第一内存区域的区域单元总个数为 $2N$;

获取所述第一内存区域的地址;

申请第二内存区域, 并将所述第一内存区域的地址存入所述第二内存区域中的第一个
5 区域单元; 其中, 所述第二内存区域的区域单元总个数为 $N+1$;

获取所述第二内存区域的地址;

申请第三内存区域, 并获取所述第三内存区域中的目标存储区域, 将所述第二内存区域的地址存入所述目标存储区域;

基于所述目标存储区域存储的所述第二内存区域的地址, 执行栈帧信息获取功能模块
10 `getStackTrace` 对应的代码段, 获取所述待处理调用栈中各栈帧的信息。

第二方面, 本公开实施例还提供了一种用于获取调用栈中栈帧信息的装置, 包括:

调用栈获取模块, 用于获取程序运行过程中存在异常的待处理调用栈; 其中, 所述待处理调用栈的每个区域单元中包括方法指针以及与所述方法指针对应的指令偏移值, 所述待处理调用栈的区域单元总个数为 N ;

15 第一内存申请模块, 用于申请第一内存区域; 其中, 所述第一内存区域的前半部分区域单元用于存储所述方法指针, 所述第一内存区域的后半部分区域单元用于存储与所述方法指针对应的指令偏移值, 所述第一内存区域的区域单元总个数为 $2N$;

第一地址获取模块, 用于获取所述第一内存区域的地址;

第二内存申请模块, 用于申请第二内存区域, 并将所述第一内存区域的地址存入所述
20 第二内存区域中的第一个区域单元; 其中, 所述第二内存区域的区域单元总个数为 $N+1$;

第二地址获取模块, 用于获取所述第二内存区域的地址;

第三内存申请模块, 用于申请第三内存区域, 并获取所述第三内存区域中的目标存储区域, 将所述第二内存区域的地址存入所述目标存储区域;

栈帧信息获取模块, 用于基于所述目标存储区域存储的所述第二内存区域的地址, 执
25 行栈帧信息获取功能模块 `getStackTrace` 对应的代码段, 获取所述待处理调用栈中各栈帧的信息。

第三方面, 本公开实施例还提供了一种电子设备, 包括存储器和处理器, 其中, 所述存储器中存储有计算机程序, 当所述计算机程序被所述处理器执行时, 使得所述电子设备实现本公开实施例提供的任一所述的用于获取调用栈中栈帧信息的方法。

30 第四方面, 本公开实施例还提供了一种计算机可读存储介质, 所述存储介质中存储有计算机程序, 当所述计算机程序被计算设备执行时, 使得所述计算设备实现本公开实施例提供的任一所述的用于获取调用栈中栈帧信息的方法。

本公开实施例提供的技术方案与现有技术相比至少具有如下优点：

在本公开实施例中，通过申请第一内存区域，将程序运行过程中存在异常的待处理调用栈中的方法指针存储在第一内存区域的前半部分区域单元，将与方法指针对应的指令偏移值存储在第一内存区域的后半部分区域单元，然后将第一内存区域的地址存储在申请的
5 第二内存区域中的第一个区域单元，其次将第二内存区域的地址存储在申请的第三内存区域中的目标存储区域，从而实现对待处理调用栈中方法指针和对应的指令偏移值的可控式存储，最后基于目标存储区域存储的第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息，解决了现有方案中无法稳定地获取栈帧的详细信息的问题，实现了稳定地获取栈帧的详细信息，并且降低了栈帧
10 详细信息获取的复杂度，对不同的系统版本具有较高的兼容性。通过获取栈帧的详细信息，有助于后续便捷定位程序运行过程中的异常，提高了解决程序异常的效率。

附图说明

此处的附图被并入说明书中并构成本说明书的一部分，示出了符合本公开的实施例，
15 并与说明书一起用于解释本公开的原理。

为了更清楚地说明本公开实施例或现有技术中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，对于本领域普通技术人员而言，在不付出创造性劳动性的前提下，还可以根据这些附图获得其他的附图。

图 1 为本公开实施例提供的一种用于获取调用栈中栈帧信息的方法的流程图；
20 图 2 为本公开实施例提供的另一种用于获取调用栈中栈帧信息的方法的流程图；
图 3 为本公开实施例提供的一种栈帧信息获取过程中各个数组之间的关系示意图；
图 4 为本公开实施例提供的一种用于获取调用栈中栈帧信息的装置的结构示意图；
图 5 为本公开实施例提供的一种电子设备的结构示意图。

25 具体实施方式

为了能够更清楚地理解本公开的上述目的、特征和优点，下面将对本公开的方案进行进一步描述。需要说明的是，在不冲突的情况下，本公开的实施例及实施例中的特征可以相互组合。

在下面的描述中阐述了很多具体细节以便于充分理解本公开，但本公开还可以采用其他
30 他不同于在此描述的方式来实施；显然，说明书中的实施例只是本公开的一部分实施例，而不是全部的实施例。

图 1 为本公开实施例提供的一种用于获取调用栈中栈帧信息的方法的流程图，该方法

可以由用于获取调用栈中栈帧信息的装置执行，该装置可以采用软件和/或硬件实现，并可集成在具有计算能力的电子设备上，例如移动终端、笔记本、电脑等。

如图 1 所示，本公开实施例提供的用于获取调用栈中栈帧信息的方法可以包括：

S101、获取程序运行过程中存在异常的待处理调用栈；其中，待处理调用栈的每个区域单元中包括方法指针以及与方法指针对应的指令偏移值，待处理调用栈的区域单元总个数为 N 。

本公开实施例中提及的程序运行过程可以是电子设备上至少一个应用程序的运行过程，可以利用现有方案中可用的调用栈获取方式获取程序运行过程中存在异常的调用栈，例如程序崩溃、程序卡顿等过程中产生的调用栈，作为后续的处理对象。

具体地，调用栈在代码底层上可以使用数组的形式表示，数组中的元素位置对应调用栈的区域单元，调用栈的区域单元总个数 N 即数组的长度 N ， N 取整数，数组的每个元素由一个方法指针 `ArtMethod`（或称为 `ArtMethod` 指针）和一个指令偏移值 `DexPc` 构成，方法指针和指令偏移值存在一一对应关系。

可选地，获取程序运行过程中存在异常的调用栈，包括：

获取程序运行过程中存在异常的多个调用栈；

通过对比各个调用栈中的方法指针和指令偏移值，确定多个调用栈中的重复调用栈；即方法指针和对应的指令偏移值均相同的调用栈，属于重复的调用栈；

基于重复调用栈对多个调用栈进行去重处理，得到待处理调用栈。

通过对获取的多个调用栈进行去重处理，可以减少待处理的调用栈的数量，提高栈帧信息获取的效率，而且不会影响获取的栈帧信息的准确性。

S102、申请第一内存区域；其中，第一内存区域的前半部分区域单元用于存储方法指针，第一内存区域的后半部分区域单元用于存储与方法指针对应的指令偏移值，第一内存区域的区域单元总个数为 $2N$ 。

通过申请第一内存区域，用于实现对待处理的每个调用栈中方法指针和指令偏移值的二次存储。具体地，方法指针可以用于获取栈帧的方法名、栈帧方法所在类名、栈帧对应的源码文件名中的至少之一；指令偏移值可以用于获取栈帧对应的源码行号。

示例性地，申请第一内存区域，包括：

为属于整型（`int` 型）或者长整型（`long` 型）的第一数组申请第一内存区域。在构建程序代码过程中，构建用于为前述第一数组申请第一内存区域的代码，例如可以使用 `new int[]` 命令或者 `new long[]` 命令，实现创建属于整型或者长整型的第一数组；其中，第一数组的元素分别存储在第一内存区域的区域单元中。针对 32 位的系统，第一数组的类型采用整型；针对 64 位的系统，第一数组的类型采用长整型。

为了确保后续稳定且准确地获取栈帧信息，方法指针或者指令偏移值在待处理调用栈中的出现顺序与在第一内存区域中的出现顺序相同，也即方法指针或者指令偏移值在待处理调用栈中的出现顺序与在第一数组中的出现顺序相同。

S103、获取第一内存区域的地址。

- 5 内存区域的地址获取可以基于任意可用的现有地址获取方式实现。第一内存区域的地址可以用于获取第一内存区域中的存储内容。

S104、申请第二内存区域，并将第一内存区域的地址存入第二内存区域中的第一个区域单元；其中，第二内存区域的区域单元总个数为 $N+1$ 。

- 10 在执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段的过程中，只关注第二内存区域中的第一个区域单元中的存储内容，因此，将第一内存区域的地址存入第二内存区域中的第一个区域单元，针对第二内存区域中其他剩余区域单元的存储内容，本公开实施例不作具体限定，例如可以保留默认值 `null`。并且，基于栈帧信息获取函数 `getStackTrace` 的本身特性，为了确保函数调用的成功实现，第二内存区域的区域单元总个数需要为 $N+1$ 。

- 15 示例性地，申请第二内存区域，包括：为对象基类 `Object` 下的第二数组申请第二内存区域。在构建程序代码过程中，构建用于为对象基类 `Object` 下的第二数组（即针对 `java` 程序，第二数组属于 `Object[]` 类型）申请第二内存区域的代码，例如可以使用 `new Object[]` 命令，实现创建对象基类 `Object` 下的第二数组；其中，第二数组的元素分别存储在第二内存区域的区域单元中。

- 20 需要说明的是，在本公开实施例中，在确保方案逻辑合理的情况下，各个操作步骤之间的执行顺序并无严格限定。例如，针对操作 S103 和操作 S104，可以先获取第一内存区域的地址，然后申请第二内存区域，还可以先申请第二内存区域，然后获取第一内存区域的地址，当然，获取第一内存区域的地址和申请第二内存区域，还可以同时执行，本公开实施例对此不作具体限定。

S105、获取第二内存区域的地址。

- 25 第二内存区域的地址可以用于获取第二内存区域中的存储内容。

S106、申请第三内存区域，并获取第三内存区域中的目标存储区域，将第二内存区域的地址存入目标存储区域。

- 30 第三内存区域中的目标存储区域可以根据栈帧信息获取函数 `getStackTrace` 的执行需求来确定。示例性地，申请第三内存区域，包括：为异常父类 `Throwable` 下的目标对象申请第三内存区域。在构建程序代码过程中，构建用于为异常父类 `Throwable` 下的目标对象申请第三内存区域的代码，例如可以使用 `new Throwable()` 命令，实现创建异常父类 `Throwable` 下的目标对象，该目标对象可以示例性地采用 `thr` 表示。完成第三内存区域的申请后，可以

获取第三内存区域中的目标存储区域，将第二内存区域的地址存入目标存储区域。S107、基于目标存储区域存储的第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息。

具体地，在 `getStackTrace` 函数执行过程中，可以通过第二内存区域的地址，获取到第二内存区域中的第一个区域单元中存储的第一内存区域的地址，进而可以通过第一内存区域的地址，获取到第一内存区域中存储的方法指针以及方法指针对应的指令偏移值，然后可以基于方法指针获取栈帧的方法名、栈帧方法所在类名、栈帧对应的源码文件名中的至少之一，基于方法指针对应的指令偏移值获取栈帧对应的源码行号。本公开实施例中提及的各栈帧的信息可以包括以下至少之一：栈帧的方法名、栈帧方法所在类名、栈帧对应的源码文件名和栈帧对应的源码行号。

示例性地，在创建异常父类 `Throwable` 下的目标对象的过程中，会同时创建目标对象的初始成员（`backtrace` 成员），目标存储区域即用于存储 `backtrace` 成员。栈帧信息获取功能模块 `getStackTrace` 对应的代码段中的 `getStackTrace` 函数属于异常父类 `Throwable` 下的目标对象的函数。通过将第二内存区域的地址存入目标存储区域，也即实现利用对象基类 `Object` 下的第二数组替换目标对象的初始成员，从而实现对 `getStackTrace` 函数的输入的有效控制，达到获取所需的栈帧信息的效果。

可选地，基于目标存储区域存储的第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息，包括：

基于目标存储区域存储的第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息的存储地址；
基于存储地址，获取各栈帧的信息。

函数 `getStackTrace` 属于系统函数，在函数执行过程中，其返回值为一个指向特定数组的存储地址，通过该存储地址可以获取到需求的栈帧信息。示例性地，该存储地址指向栈帧信息数组 `StackTraceElement`，栈帧信息数组中的每个元素对应不同栈帧的信息。

例如，获取栈帧信息数组 `StackTraceElement` 中的每个元素对应的不同栈帧的信息，可以采用以下程序命令实现：

```
StackTraceElement[] elements = thr. getStackTrace (); //thr 用于指代异常父类 Throwable 下的目标对象；elements 作为 StackTraceElement[] 类型下的一个数组名，此处仅用于示例，可以灵活确定数组名
```

```
elements[0].getMethodName (); //当前栈帧的方法名  
elements[0].getClassName (); //当前栈帧方法所在类名  
elements[0].getFileName (); //当前栈帧对应的源码文件名
```

`elements[0].getLineNumber (); //当前栈帧对应的源码行号`

以上程序命令作为一种示例，不应理解为对本公开实施例的具体限定。

为便于理解本公开实施例，以下从程序编写的角度，利用示例性地程序代码对本公开实施例进行说明，但不应理解为对本公开实施例的具体限定。具体地，以待处理调用栈的区域单元总个数 N 等于 2 为例，进行示例性说明，即待处理调用栈中包括 2 组元素：方法指针 `method1` 和指令偏移值 `dexpc1`、方法指针 `method2` 和指令偏移 `dexpc2`。程序代码示例如下：

`int method1, method2, dexpc1, dexpc2; //本方案的原始输入，即待处理调用栈中的方法指针和指令偏移值`

`method1=973; //以下关于 method1、method1 、dexpc1、dexpc2 的具体取值，需要根据实际获取的调用栈而定，此处的各个数值仅作为一种示例`

`method2=460;`

`dexpc1=0;`

`dexpc2=0;`

`int[] firstIL=new int [4]; //创建属于整型且长度为4(即元素个数为4)的第一数组 firstIL, firstIL 作为数组名只是一种示例，可以根据程序编写情况灵活设置，并且，此处以 32 位系统为例，创建整型数组，如果是 64 位系统，则需创建长整型数组`

`firstIL[0]= method1; //将调用栈中的第一个方法指针 method1 作为数组 firstIL 的第一个元素`

`firstIL[1]= method2; //将调用栈中的第二个方法指针 method2 作为数组 firstIL 的第二个元素`

`firstIL[2]= dexpc1; //将调用栈中的第一个指令偏移 dexpc1 作为数组 firstIL 的第三个元素`

`firstIL[3]= dexpc2; //将调用栈中的第二个指令偏移 dexpc2 作为数组 firstIL 的第四个元素`

`Object[] backtrace=new Object[3]; //创建对象基类 Object 下的第二数组 backtrace, backtrace 作为数组名只是一种示例，可以根据程序编写情况灵活设置，第二数组 backtrace 的长度为 3，即元素个数为 3`

`backtrace[0]= firstIL; //将第一数组 firstIL 的指针作为第二数组 backtrace 的第一元素，第二数组 backtrace 的其他元素可以取默认值 null`

Throwable thr=new Throwable (); //创建异常父类 Throwable 下的目标对象 thr, thr 作为对象名只是一种示例, 可以根据程序编写情况灵活设置

```

try {
5      Field backtraceFiled= Throwable.class.getDeclaredField (name: "backtrace");
      backtraceFiled.set (thr, backtrace); //利用对象基类 Object 下的第二数组 backtrace
      替换目标对象 thr 的初始成员 backtrace
      StackTraceElement[] elements = thr. getStackTrace (); //调用系统函数 getStackTrace 获
      取调用栈中各个栈帧的信息
10     } catch (Throwable e) {
        e.ptintStackTrace (); //输出各个栈帧的信息
    }

```

进一步地, 本公开实施例提供的方法还可以包括: 将各栈帧的信息发送至服务端; 其中, 服务端用于基于各栈帧的信息分析程序运行过程中的异常, 从而提高解决程序异常的效率。示例性地, 服务器可以与多个用于执行本公开实施例技术方案的电子设备进行通信, 基于针对相同应用程序运行过程中的调用栈中各栈帧的信息, 通过信息的统计与分析, 准确定位应用程序运行过程中的异常。

在本公开实施例中, 通过申请第一内存区域, 将待处理调用栈中的方法指针存储在第一内存区域的前半部分区域单元, 将与方法指针对应的指令偏移值存储在第一内存区域的后半部分区域单元, 然后将第一内存区域的地址存储在申请的第二内存区域中的第一个区域单元, 其次将第二内存区域的地址存储在申请的第三内存区域中的目标存储区域, 从而实现对待处理调用栈中方法指针和对应的指令偏移值的可控式存储, 最后基于目标存储区域存储的第二内存区域的地址, 执行栈帧信息获取功能模块 getStackTrace 对应的代码段, 获取待处理调用栈中各栈帧的信息, 解决了现有方案中无法稳定地获取栈帧的详细信息的问题, 实现了稳定地获取栈帧的详细信息, 并且降低了栈帧详细信息获取的复杂度, 对不同的系统版本具有较高的兼容性。通过获取栈帧的详细信息, 有助于后续便捷定位程序运行过程中的异常, 提高了解决程序异常的效率。

图 2 为本公开实施例提供的另一种用于获取调用栈中栈帧信息的方法的流程图, 用于对本公开实施例进行示例性说明, 不应理解为对本公开实施例的具体限定, 具体地, 图 2 从计算机程序执行流程的角度对本公开实施例进行示例性解释。

如图 2 所示, 在程序执行过程中, 可以首先获取到程序运行过程中存在异常的调用栈

的底层表示：例如一个数组，该数组可以在编程阶段便示例性地定义为采用为 rawArr 表示，数组的每个元素由一个 ArtMethod 指针和一个对应的指令偏移值 DexPc 构成，数组长度记为 N；创建一个长度为 2N 的 int 或 long 型数组（32 位系统情况下是 int 型，64 位系统情况下是 long 型），该数组可以在编程阶段便示例性地定义为采用 firstIL 表示；

5 然后，将数组 rawArr 中的所有 ArtMethod 指针设置为数组 firstIL 的前半段元素，将数组 rawArr 的所有 DexPc 值设置为数组 firstIL 的后半段元素，其中由于数组 firstIL 的长度是数组 rawArr 长度的两倍，所以 ArtMethod 指针和 DexPc 值分别占用数组 firstIL 的一半，并且，ArtMethod 指针或 DexPc 值在数组 firstIL 中的出现顺序在数组 rawArr 中的出现顺序相同；

10 创建一个 Object[] 类型（对象基类）的数组，该数组可以在编程阶段便示例性地定义为采用 bt 表示，其长度为 N+1，然后令数组 bt 的第一个元素为数组 firstIL，数组 bt 剩余的 N 个元素保留默认值 null 即可；图 3 作为示例，示出了各个数组之间的关系示意图；

 创建一个 Throwable 类（异常父类）的对象，该对象可以在编程阶段便示例性地定义为采用 thr 表示，利用反射将对象 thr 的 backtrace 成员替换为上述数组 bt，然后调用对象 thr
15 的 getStackTrace 函数，得到 StackTraceElement 类型的数组，该数组中的每个元素对应一个栈帧，并提供了栈帧的方法名、栈帧方法所在类名、栈帧对应的源码文件名和栈帧对应的源码行号等详细信息。

 本公开实施例实现了稳定地获取栈帧的详细信息，并且降低了栈帧详细信息获取的复杂度，对不同的系统版本具有较高的兼容性。

20 图 4 为本公开实施例提供的一种用于获取调用栈中栈帧信息的装置的结构示意图，该装置可以采用软件和/或硬件实现，并可集成在任意具有计算能力的电子设备上，例如移动终端、笔记本、电脑等。

 如图 4 所示，本公开实施例提供的用于获取调用栈中栈帧信息的装置 400 可以包括调用栈获取模块 401、第一内存申请模块 402、第一地址获取模块 403、第二内存申请模块 404、
25 第二地址获取模块 405、第三内存申请模块 406 和栈帧信息获取模块 407，其中：

 调用栈获取模块 401，用于获取程序运行过程中存在异常的待处理调用栈；其中，待处理调用栈的每个区域单元中包括方法指针以及与方法指针对应的指令偏移值，待处理调用栈的区域单元总个数为 N；

 第一内存申请模块 402，用于申请第一内存区域；其中，第一内存区域的前半部分区域单元用于存储方法指针，第一内存区域的后半部分区域单元用于存储与方法指针对应的指令偏移值，第一内存区域的区域单元总个数为 2N；
30

 第一地址获取模块 403，用于获取第一内存区域的地址；

第二内存申请模块 404，用于申请第二内存区域，并将第一内存区域的地址存入第二内存区域中的第一个区域单元；其中，第二内存区域的区域单元总个数为 $N+1$ ；

第二地址获取模块 405，用于获取第二内存区域的地址；

5 第三内存申请模块 406，用于申请第三内存区域，并获取第三内存区域中的目标存储区域，将第二内存区域的地址存入目标存储区域；

栈帧信息获取模块 407，用于基于目标存储区域存储的第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息。

可选地，第一内存申请模块 402 具体用于：

10 为属于整型或者长整型的第一数组申请第一内存区域；其中，第一数组的元素分别存储在第一内存区域的区域单元中；

可选地，第二内存申请模块 404 具体用于：

为对象基类 `Object` 下的第二数组申请第二内存区域；其中，第二数组的元素分别存储在第二内存区域的区域单元中；

可选地，第三内存申请模块 406 具体用于：

15 为异常父类 `Throwable` 下的目标对象申请第三内存区域。

可选地，方法指针或者指令偏移值在待处理调用栈中的出现顺序与在第一内存区域中的出现顺序相同。

可选地，栈帧信息获取模块 407 包括：

20 存储地址返回单元，用于基于目标存储区域存储的第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息的存储地址；

栈帧信息获取单元，用于基于存储地址，获取各栈帧的信息。

可选地，存储地址指向栈帧信息数组，栈帧信息数组中的每个元素对应不同栈帧的信息。

25 可选地，各栈帧的信息包括以下至少之一：栈帧的方法名、栈帧方法所在类名、栈帧对应的源码文件名和栈帧对应的源码行号。

可选地，本公开实施例提供的用于获取调用栈中栈帧信息的装置 400 还包括：

栈帧信息发送模块，用于将各栈帧的信息发送至服务端；其中，服务端用于基于各栈帧的信息分析程序运行过程中的异常。

30 可选地，调用栈获取模块 401 包括：

调用栈获取单元，用于获取程序运行过程中存在异常的多个调用栈；

重复调用栈确定单元，用于通过对比各个调用栈中的方法指针和指令偏移值，确定多

个调用栈中的重复调用栈；

调用栈去重单元，用于基于重复调用栈对多个调用栈进行去重处理，得到待处理调用栈。

本公开实施例所提供的用于获取调用栈中栈帧信息的装置可执行本公开实施例所提供的任意用于获取调用栈中栈帧信息的方法，具备执行方法相应的功能模块和有益效果。本公开装置实施例中未详尽描述的内容可以参考本公开任意方法实施例中的描述。

图 5 为本公开实施例提供的一种电子设备的结构示意图，用于对实现本公开实施例提供的用于获取调用栈中栈帧信息的方法的电子设备进行示例性说明。本公开实施例中的电子设备可以包括但不限于诸如移动电话、笔记本电脑、数字广播接收器、PDA（个人数字助理）、PAD（平板电脑）、PMP（便携式多媒体播放器）、车载终端（例如车载导航终端）等等的移动终端以及诸如数字 TV、台式计算机、智能家居设备、可穿戴电子设备、服务器等等的固定终端。图 5 示出的电子设备仅仅是一个示例，不应对本公开实施例的功能和占用范围带来任何限制。

如图 5 所示，电子设备 500 包括一个或多个处理器 501 和存储器 502。处理器 501 可以是中央处理单元（CPU）或者具有数据处理能力和/或指令执行能力的其他形式的处理单元，并且可以控制电子设备 500 中的其他组件以执行期望的功能。

存储器 502 可以包括一个或多个计算机程序产品，计算机程序产品可以包括各种形式的计算机可读存储介质，例如易失性存储器和/或非易失性存储器。易失性存储器例如可以包括随机存取存储器（RAM）和/或高速缓冲存储器（cache）等。非易失性存储器例如可以包括只读存储器（ROM）、硬盘、闪存等。在计算机可读存储介质上可以存储一个或多个计算机程序指令，处理器 501 可以运行程序指令，以实现本公开实施例提供的用于获取调用栈中栈帧信息的方法，还可以实现其他期望的功能。在计算机可读存储介质中还可以存储诸如输入信号、信号分量、噪声分量等各种内容。

其中，本公开实施例提供的用于获取调用栈中栈帧信息的方法可以包括：获取程序运行过程中存在异常的待处理调用栈；其中，待处理调用栈的每个区域单元中包括方法指针以及与方法指针对应的指令偏移值，待处理调用栈的区域单元总个数为 N ；申请第一内存区域；其中，第一内存区域的前半部分区域单元用于存储方法指针，第一内存区域的后半部分区域单元用于存储与方法指针对应的指令偏移值，第一内存区域的区域单元总个数为 $2N$ ；获取第一内存区域的地址；申请第二内存区域，并将第一内存区域的地址存入第二内存区域中的第一个区域单元；其中，第二内存区域的区域单元总个数为 $N+1$ ；获取第二内存区域的地址；申请第三内存区域，并获取第三内存区域中的目标存储区域，将第二内存区域的地址存入目标存储区域；基于目标存储区域存储的第二内存区域的地址，执行栈帧

信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息。应当理解，电子设备 500 还可以执行本公开方法实施例提供的其他可选实施方案。

在一个示例中，电子设备 500 还可以包括：输入装置 503 和输出装置 504，这些组件通过总线系统和/或其他形式的连接机构（未示出）互连。

5 此外，该输入装置 503 还可以包括例如键盘、鼠标等等。

该输出装置 504 可以向外部输出各种信息，包括确定出的距离信息、方向信息等。该输出装置 504 可以包括例如显示器、扬声器、打印机、以及通信网络及其所连接的远程输出设备等等。

当然，为了简化，图 5 中仅示出了该电子设备 500 中与本公开有关的组件中的一些，
10 省略了诸如总线、输入/输出接口等等的组件。除此之外，根据具体应用情况，电子设备 500 还可以包括任何其他适当的组件。

除了上述方法和设备以外，本公开实施例还提供一种计算机程序产品，其包括计算机程序或计算机程序指令，计算机程序或计算机程序指令在被计算设备执行时使得计算设备实现本公开实施例所提供的任意用于获取调用栈中栈帧信息的方法。

15 计算机程序产品可以以一种或多种程序设计语言的任意组合来编写用于执行本公开实施例操作的程序代码，程序设计语言包括面向对象的程序设计语言，诸如 Java、C++等，还包括常规的过程式程序设计语言，诸如“C”语言或类似的设计语言。程序代码可以完全地在用户电子设备上执行、部分地在用户电子设备上执行、作为一个独立的软件包执行、部分在用户电子设备上且部分在远程电子设备上执行、或者完全在远程电子设备上执
20 行。

此外，本公开实施例还可以提供一种计算机可读存储介质，其上存储有计算机程序指令，计算机程序指令在被计算设备执行时使得计算设备实现本公开实施例所提供的任意用于获取调用栈中栈帧信息的方法。

其中，本公开实施例提供的用于获取调用栈中栈帧信息的方法可以包括：获取程序运行过程中存在异常的待处理调用栈；其中，待处理调用栈的每个区域单元中包括方法指针
25 以及与方法指针对应的指令偏移值，待处理调用栈的区域单元总个数为 N ；申请第一内存区域；其中，第一内存区域的前半部分区域单元用于存储方法指针，第一内存区域的后半部分区域单元用于存储与方法指针对应的指令偏移值，第一内存区域的区域单元总个数为 $2N$ ；获取第一内存区域的地址；申请第二内存区域，并将第一内存区域的地址存入第二内存区域中的第一个区域单元；其中，第二内存区域的区域单元总个数为 $N+1$ ；获取第二内存区域的地址；申请第三内存区域，并获取第三内存区域中的目标存储区域，将第二内存区域的地址存入目标存储区域；基于目标存储区域存储的第二内存区域的地址，执行栈帧
30

信息获取功能模块 `getStackTrace` 对应的代码段，获取待处理调用栈中各栈帧的信息。应当理解，计算机程序指令在被计算设备执行时，还可以使得计算设备实现本公开方法实施例提供的其他可选实施方案。

计算机可读存储介质可以采用一个或多个可读介质的任意组合。可读介质可以是可读信号介质或者可读存储介质。可读存储介质例如可以包括但不限于电、磁、光、电磁、红外线、或半导体的系统、装置或器件，或者任意以上的组合。可读存储介质的更具体的例子（非穷举的列表）包括：具有一个或多个导线的电连接、便携式盘、硬盘、随机存取存储器（RAM）、只读存储器（ROM）、可擦式可编程只读存储器（EPROM 或闪存）、光纤、便携式紧凑盘只读存储器(CD-ROM)、光存储器件、磁存储器件、或者上述的任意合适的组合。

需要说明的是，在本文中，诸如“第一”和“第二”等之类的关系术语仅仅用来将一个实体或者操作与另一个实体或操作区分开来，而不一定要求或者暗示这些实体或操作之间存在任何这种实际的关系或者顺序。而且，术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含，从而使得包括一系列要素的过程、方法、物品或者设备不仅包括那些要素，而且还包括没有明确列出的其他要素，或者是还包括为这种过程、方法、物品或者设备所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括要素的过程、方法、物品或者设备中还存在另外的相同要素。

以上仅是本公开的具体实施方式，使本领域技术人员能够理解或实现本公开。对这些实施例的多种修改对本领域的技术人员来说将是显而易见的，本文中所定义的一般原理可以在不脱离本公开的精神或范围的情况下，在其它实施例中实现。因此，本公开将不会被限制于本文的这些实施例，而是要符合与本文所公开的原理和新颖特点相一致的最宽的范围。

权利要求书

1、一种用于获取调用栈中栈帧信息的方法，其特征在于，包括：

获取程序运行过程中存在异常的待处理调用栈；其中，所述待处理调用栈的每个区域单元中包括方法指针以及与所述方法指针对应的指令偏移值，所述待处理调用栈的区域单元总个数为 N ；

申请第一内存区域；其中，所述第一内存区域的前半部分区域单元用于存储所述方法指针，所述第一内存区域的后半部分区域单元用于存储与所述方法指针对应的指令偏移值，所述第一内存区域的区域单元总个数为 $2N$ ；

获取所述第一内存区域的地址；

申请第二内存区域，并将所述第一内存区域的地址存入所述第二内存区域中的第一个区域单元；其中，所述第二内存区域的区域单元总个数为 $N+1$ ；

获取所述第二内存区域的地址；

申请第三内存区域，并获取所述第三内存区域中的目标存储区域，将所述第二内存区域的地址存入所述目标存储区域；

基于所述目标存储区域存储的所述第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取所述待处理调用栈中各栈帧的信息。

2、根据权利要求 1 所述的方法，其特征在于：

申请第一内存区域，包括：

为属于整型或者长整型的第一数组申请所述第一内存区域；其中，所述第一数组的元素分别存储在所述第一内存区域的区域单元中；

申请第二内存区域，包括：

为对象基类 `Object` 下的第二数组申请所述第二内存区域；其中，所述第二数组的元素分别存储在所述第二内存区域的区域单元中；

申请第三内存区域，包括：

为异常父类 `Throwable` 下的目标对象申请所述第三内存区域。

3、根据权利要求 1 所述的方法，其特征在于，所述方法指针或者所述指令偏移值在所述待处理调用栈中的出现顺序与在所述第一内存区域中的出现顺序相同。

4、根据权利要求 1 所述的方法，其特征在于，基于所述目标存储区域存储的所述第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取所述待处理调用栈中各栈帧的信息，包括：

基于所述目标存储区域存储的所述第二内存区域的地址，执行所述栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取所述待处理调用栈中各栈帧的信息的存储地址；

基于所述存储地址，获取所述各栈帧的信息。

5、根据权利要求1所述的方法，其特征在于，所述各栈帧的信息包括以下至少之一：栈帧的方法名、栈帧方法所在类名、栈帧对应的源码文件名和栈帧对应的源码行号。

6、根据权利要求1所述的方法，其特征在于，还包括：

5 将所述各栈帧的信息发送至服务端；其中，所述服务端用于基于所述各栈帧的信息分析程序运行过程中的异常。

7、根据权利要求1所述的方法，其特征在于，获取程序运行过程中存在异常的调用栈，包括：

获取程序运行过程中存在异常的多个调用栈；

10 通过对比各个调用栈中的所述方法指针和所述指令偏移值，确定所述多个调用栈中的重复调用栈；

基于所述重复调用栈对所述多个调用栈进行去重处理，得到所述待处理调用栈。

8、一种用于获取调用栈中栈帧信息的装置，其特征在于，包括：

15 调用栈获取模块，用于获取程序运行过程中存在异常的待处理调用栈；其中，所述待处理调用栈的每个区域单元中包括方法指针以及与所述方法指针对应的指令偏移值，所述待处理调用栈的区域单元总个数为 N ；

第一内存申请模块，用于申请第一内存区域；其中，所述第一内存区域的前半部分区域单元用于存储所述方法指针，所述第一内存区域的后半部分区域单元用于存储与所述方法指针对应的指令偏移值，所述第一内存区域的区域单元总个数为 $2N$ ；

第一地址获取模块，用于获取所述第一内存区域的地址；

20 第二内存申请模块，用于申请第二内存区域，并将所述第一内存区域的地址存入所述第二内存区域中的第一个区域单元；其中，所述第二内存区域的区域单元总个数为 $N+1$ ；

第二地址获取模块，用于获取所述第二内存区域的地址；

第三内存申请模块，用于申请第三内存区域，并获取所述第三内存区域中的目标存储区域，将所述第二内存区域的地址存入所述目标存储区域；

25 栈帧信息获取模块，用于基于所述目标存储区域存储的所述第二内存区域的地址，执行栈帧信息获取功能模块 `getStackTrace` 对应的代码段，获取所述待处理调用栈中各栈帧的信息。

9、一种电子设备，其特征在于，包括存储器和处理器，其中，所述存储器中存储有计算机程序，当所述计算机程序被所述处理器执行时，使得所述电子设备实现权利要求 1-7 30 中任一项所述的用于获取调用栈中栈帧信息的方法。

10、一种计算机可读存储介质，其特征在于，所述存储介质中存储有计算机程序，当所述计算机程序被计算设备执行时，使得所述计算设备实现权利要求 1-7 中任一项所述的

用于获取调用栈中栈帧信息的方法。

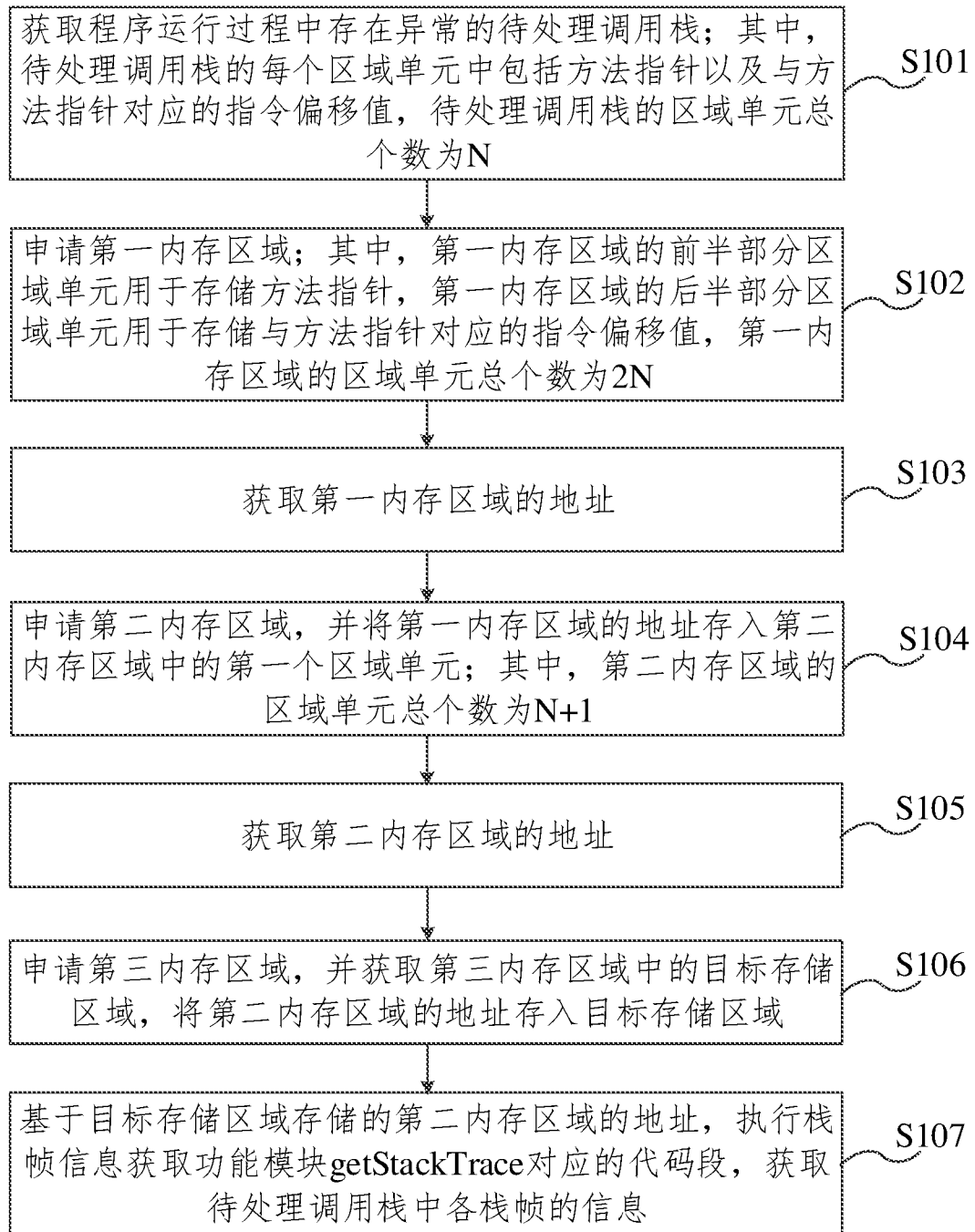


图 1

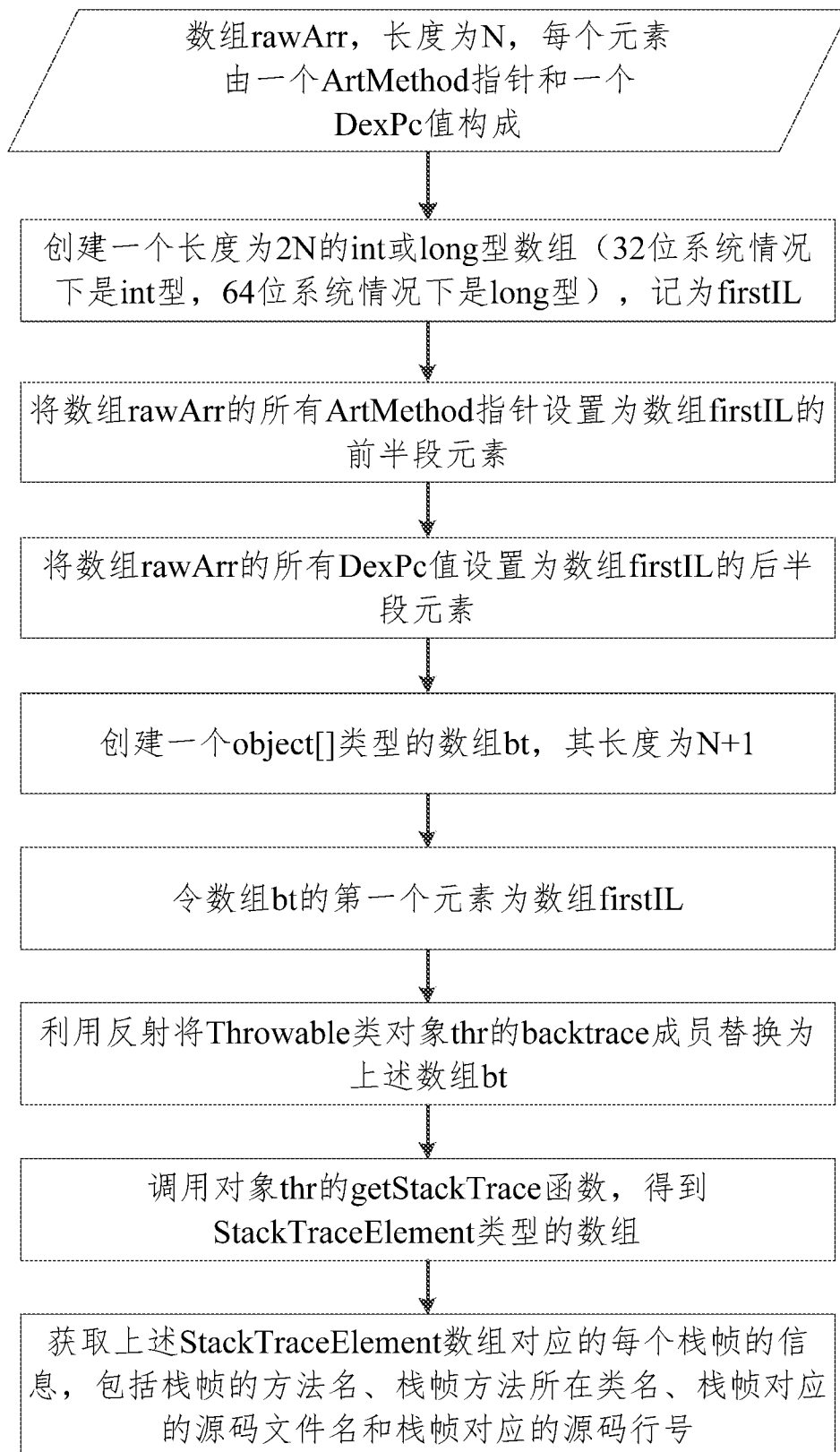


图 2

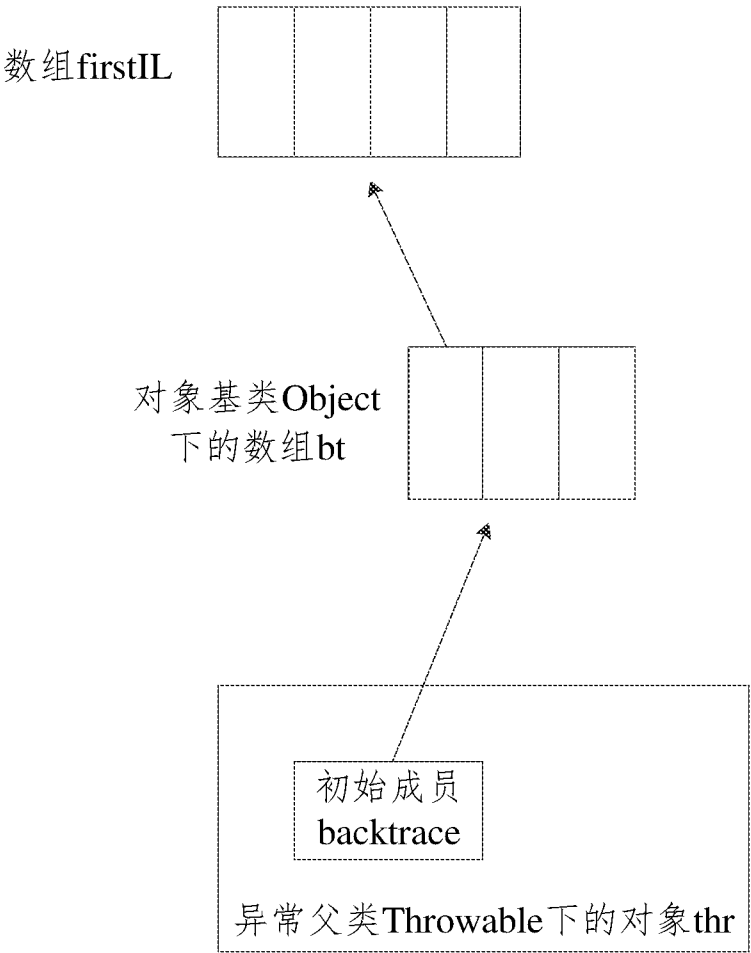


图 3

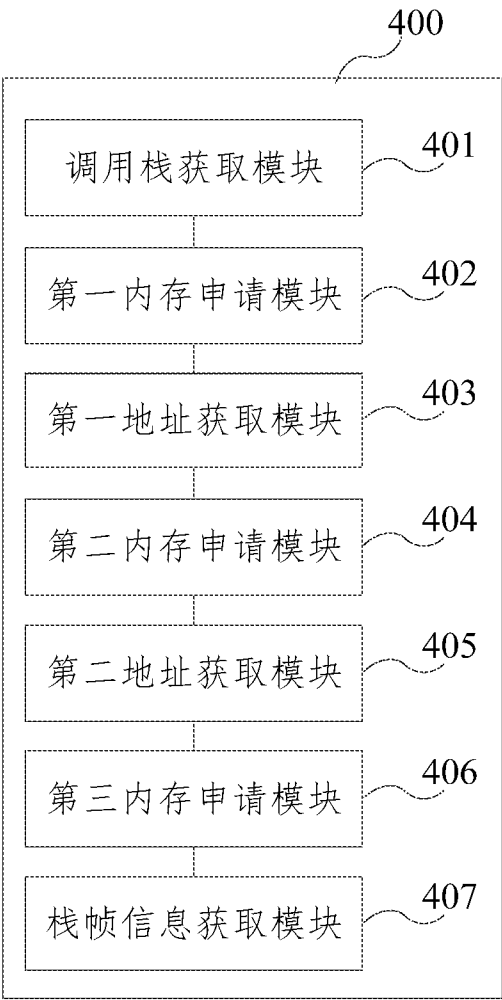


图 4

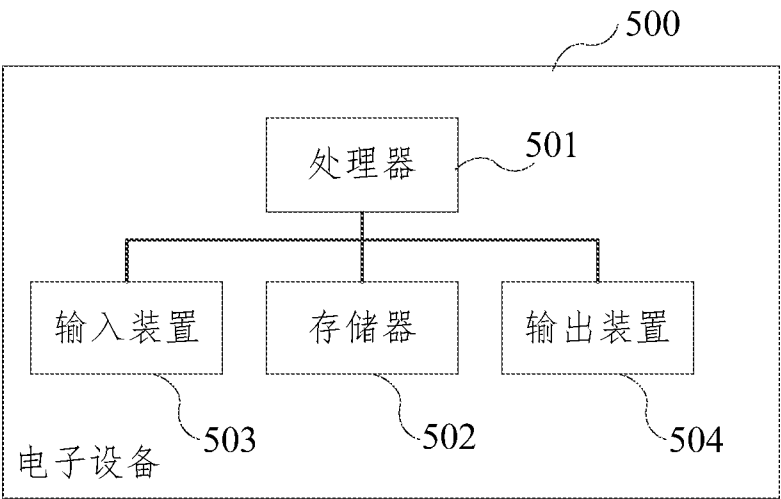


图 5

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2022/078365

A. CLASSIFICATION OF SUBJECT MATTER

G06F 11/07(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNABS; CNTXT; WPABSC; ENTXTC; VEN; ENTXT; CJFD; CNKI; IEEE: 栈, 调用, 异常, 错误, 指针, 地址, 偏移, 内存, 存储, stack, call, abnormal, error, fault, pointer, address, offset, memory, store

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
PX	CN 112905381 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD.) 04 June 2021 (2021-06-04) description, paragraphs 35-141	1-10
A	CN 111813465 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD.) 23 October 2020 (2020-10-23) description, paragraphs 47-92	1-10
A	CN 112084024 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD.) 15 December 2020 (2020-12-15) description, paragraphs 36-126	1-10
A	CN 103106132 A (CHINA STANDARD SOFTWARE CO., LTD.) 15 May 2013 (2013-05-15) entire document	1-10
A	US 2020371945 A1 (VDOO CONNECTED TRUST LTD.) 26 November 2020 (2020-11-26) entire document	1-10



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

“A” document defining the general state of the art which is not considered to be of particular relevance

“E” earlier application or patent but published on or after the international filing date

“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

“O” document referring to an oral disclosure, use, exhibition or other means

“P” document published prior to the international filing date but later than the priority date claimed

“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&” document member of the same patent family

Date of the actual completion of the international search

27 April 2022

Date of mailing of the international search report

09 May 2022

Name and mailing address of the ISA/CN

China National Intellectual Property Administration (ISA/
CN)
No. 6, Xitucheng Road, Jimenqiao, Haidian District, Beijing
100088, China

Facsimile No. (86-10)62019451

Authorized officer

Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2022/078365

C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	梁玉 等 (LIANG, Yu et al.). "S-Tracker: 基于栈异常的shellcode检测方法 (Non-official translation: S-Tracker: Shellcode Detection Method Based on Stack Irregularity)" 华中科技大学学报 (自然科学版) (<i>Journal of Huazhong University of Science and Technology (Natural Science Edition)</i>), Vol. 42, No. 11, 30 November 2014 (2014-11-30), ISSN: 1671-4512, sections 1-3	1-10
A	HUANG, Wei et al. "LMP:Light-Weighted Memory Protection with Hardware Assistance" ACSAC '16: 2016 Annual Computer Security Applications Conference, 09 December 2016 (2016-12-09), sections 1-5	1-10

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2022/078365

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	112905381	A	04 June 2021	None			
CN	111813465	A	23 October 2020	None			
CN	112084024	A	15 December 2020	None			
CN	103106132	A	15 May 2013	CN	103106132	B	16 March 2016
US	2020371945	A1	26 November 2020	US	11188470	B2	30 November 2021

国际检索报告

国际申请号

PCT/CN2022/078365

A. 主题的分类 G06F 11/07 (2006.01) i 按照国际专利分类 (IPC) 或者同时按照国家分类和 IPC 两种分类																							
B. 检索领域 检索的最低限度文献 (标明分类系统和分类号) G06F 包含在检索领域中的除最低限度文献以外的检索文献 在国际检索时查阅的电子数据库 (数据库的名称, 和使用的检索词 (如使用)) CNABS; CNTXT; WPABSC; ENTXTC; VEN; ENTXT; CJFD; CNKI; IEEE: 栈, 调用, 异常, 错误, 指针, 地址, 偏移, 内存, 存储, stack, call, abnormal, error, fault, pointer, address, offset, memory, store																							
C. 相关文件 <table border="1"> <thead> <tr> <th>类 型*</th> <th>引用文件, 必要时, 指明相关段落</th> <th>相关的权利要求</th> </tr> </thead> <tbody> <tr> <td>PX</td> <td>CN 112905381 A (北京字节跳动网络技术有限公司) 2021年6月4日 (2021 - 06 - 04) 说明书第35-141段</td> <td>1-10</td> </tr> <tr> <td>A</td> <td>CN 111813465 A (北京字节跳动网络技术有限公司) 2020年10月23日 (2020 - 10 - 23) 说明书第47-92段</td> <td>1-10</td> </tr> <tr> <td>A</td> <td>CN 112084024 A (北京字节跳动网络技术有限公司) 2020年12月15日 (2020 - 12 - 15) 说明书第36-126段</td> <td>1-10</td> </tr> <tr> <td>A</td> <td>CN 103106132 A (中标软件有限公司) 2013年5月15日 (2013 - 05 - 15) 全文</td> <td>1-10</td> </tr> <tr> <td>A</td> <td>US 2020371945 A1 (VD00 CONNECTED TRUST LTD) 2020年11月26日 (2020 - 11 - 26) 全文</td> <td>1-10</td> </tr> <tr> <td>A</td> <td>梁玉 等. "S-Tracker: 基于栈异常的shellcode检测方法" 《华中科技大学学报 (自然科学版)》, 第42卷, 第11期, 2014年11月30日 (2014 - 11 - 30), ISSN: 1671-4512, 第1-3节</td> <td>1-10</td> </tr> </tbody> </table>			类 型*	引用文件, 必要时, 指明相关段落	相关的权利要求	PX	CN 112905381 A (北京字节跳动网络技术有限公司) 2021年6月4日 (2021 - 06 - 04) 说明书第35-141段	1-10	A	CN 111813465 A (北京字节跳动网络技术有限公司) 2020年10月23日 (2020 - 10 - 23) 说明书第47-92段	1-10	A	CN 112084024 A (北京字节跳动网络技术有限公司) 2020年12月15日 (2020 - 12 - 15) 说明书第36-126段	1-10	A	CN 103106132 A (中标软件有限公司) 2013年5月15日 (2013 - 05 - 15) 全文	1-10	A	US 2020371945 A1 (VD00 CONNECTED TRUST LTD) 2020年11月26日 (2020 - 11 - 26) 全文	1-10	A	梁玉 等. "S-Tracker: 基于栈异常的shellcode检测方法" 《华中科技大学学报 (自然科学版)》, 第42卷, 第11期, 2014年11月30日 (2014 - 11 - 30), ISSN: 1671-4512, 第1-3节	1-10
类 型*	引用文件, 必要时, 指明相关段落	相关的权利要求																					
PX	CN 112905381 A (北京字节跳动网络技术有限公司) 2021年6月4日 (2021 - 06 - 04) 说明书第35-141段	1-10																					
A	CN 111813465 A (北京字节跳动网络技术有限公司) 2020年10月23日 (2020 - 10 - 23) 说明书第47-92段	1-10																					
A	CN 112084024 A (北京字节跳动网络技术有限公司) 2020年12月15日 (2020 - 12 - 15) 说明书第36-126段	1-10																					
A	CN 103106132 A (中标软件有限公司) 2013年5月15日 (2013 - 05 - 15) 全文	1-10																					
A	US 2020371945 A1 (VD00 CONNECTED TRUST LTD) 2020年11月26日 (2020 - 11 - 26) 全文	1-10																					
A	梁玉 等. "S-Tracker: 基于栈异常的shellcode检测方法" 《华中科技大学学报 (自然科学版)》, 第42卷, 第11期, 2014年11月30日 (2014 - 11 - 30), ISSN: 1671-4512, 第1-3节	1-10																					
☒ 其余文件在C栏的续页中列出。 ☒ 见同族专利附件。																							
<table border="0"> <tr> <td> * 引用文件的具体类型: "A" 认为不特别相关的表示了现有技术一般状态的文件 "E" 在国际申请日的当天或之后公布的在先申请或专利 "L" 可能对优先权要求构成怀疑的文件, 或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件 (如具体说明的) "O" 涉及口头公开、使用、展览或其他方式公开的文件 "P" 公布日先于国际申请日但迟于所要求的优先权日的文件 </td> <td> "T" 在申请日或优先权日之后公布, 与申请不相抵触, 但为了理解发明之理论或原理的在后文件 "X" 特别相关的文件, 单独考虑该文件, 认定要求保护的发明不是新颖的或不具有创造性 "Y" 特别相关的文件, 当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时, 要求保护的发明不具有创造性 "&" 同族专利的文件 </td> </tr> </table>			* 引用文件的具体类型: "A" 认为不特别相关的表示了现有技术一般状态的文件 "E" 在国际申请日的当天或之后公布的在先申请或专利 "L" 可能对优先权要求构成怀疑的文件, 或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件 (如具体说明的) "O" 涉及口头公开、使用、展览或其他方式公开的文件 "P" 公布日先于国际申请日但迟于所要求的优先权日的文件	"T" 在申请日或优先权日之后公布, 与申请不相抵触, 但为了理解发明之理论或原理的在后文件 "X" 特别相关的文件, 单独考虑该文件, 认定要求保护的发明不是新颖的或不具有创造性 "Y" 特别相关的文件, 当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时, 要求保护的发明不具有创造性 "&" 同族专利的文件																			
* 引用文件的具体类型: "A" 认为不特别相关的表示了现有技术一般状态的文件 "E" 在国际申请日的当天或之后公布的在先申请或专利 "L" 可能对优先权要求构成怀疑的文件, 或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件 (如具体说明的) "O" 涉及口头公开、使用、展览或其他方式公开的文件 "P" 公布日先于国际申请日但迟于所要求的优先权日的文件	"T" 在申请日或优先权日之后公布, 与申请不相抵触, 但为了理解发明之理论或原理的在后文件 "X" 特别相关的文件, 单独考虑该文件, 认定要求保护的发明不是新颖的或不具有创造性 "Y" 特别相关的文件, 当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时, 要求保护的发明不具有创造性 "&" 同族专利的文件																						
国际检索实际完成的日期		国际检索报告邮寄日期																					
2022年4月27日		2022年5月9日																					
ISA/CN的名称和邮寄地址		受权官员																					
中国国家知识产权局 (ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088 传真号 (86-10)62019451		马婷婷 电话号码 86-(20)-28958507																					

C. 相关文件		
类 型*	引用文件，必要时，指明相关段落	相关的权利要求
A	HUANG, Wei 等. "LMP:Light-Weighted Memory Protection with Hardware Assistance" 《ACSAC '16: 2016 Annual Computer Security Applications Conference》, 2016年12月9日 (2016 - 12 - 09), 第1-5节	1-10

国际检索报告
关于同族专利的信息

国际申请号

PCT/CN2022/078365

检索报告引用的专利文件			公布日 (年/月/日)	同族专利	公布日 (年/月/日)
CN	112905381	A	2021年6月4日	无	
CN	111813465	A	2020年10月23日	无	
CN	112084024	A	2020年12月15日	无	
CN	103106132	A	2013年5月15日	CN	103106132 B 2016年3月16日
US	2020371945	A1	2020年11月26日	US	11188470 B2 2021年11月30日