(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 116089912 A
(43)申请公布日 2023.05.09

(21)申请号 202211733415.0

(22)申请日 2022.12.30

(71)申请人 成都鲁易科技有限公司
地址 610095 四川省成都市中国(四川)自由贸易试验区天府新区兴隆街道湖畔路西段99号附0L-10-202106013

(72)发明人 廖恒 潘明

(74)专利代理机构 北京中强智尚知识产权代理有限公司 11448
专利代理师 王欢

(51)Int.Cl.
G06F 21/12(2013.01)
G06F 21/51(2013.01)

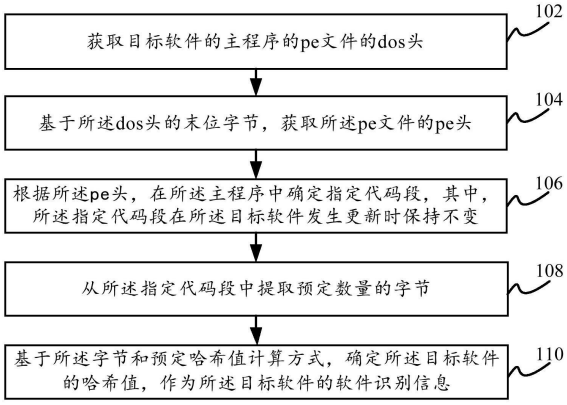
权利要求书2页 说明书9页 附图4页

(54)发明名称

软件识别信息获取方法及装置、电子设备和存储介质

(57)摘要

本申请提出了一种软件识别信息获取方法及装置、电子设备和存储介质，该方法包括：获取目标软件的pe文件的dos头；基于所述dos头的末位字节，获取所述pe文件的pe头；根据所述pe头，在所述主程序中确定指定代码段；从所述指定代码段中提取预定数量的字节；基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息。本申请的技术方案，可以在软件更新时保持软件识别信息不变，从而减少因重新生成软件识别信息而消耗的资源，降低维护软件识别信息的成本，提升了软件识别的便利性和安全性。



CN 116089912 A

1.一种软件识别信息获取方法,其特征在于,包括:

获取目标软件的pe文件的dos头;

基于所述dos头的末位字节,获取所述pe文件的pe头;

根据所述pe头,在所述主程序中确定指定代码段;

从所述指定代码段中提取预定数量的字节;

基于所述字节和预定哈希值计算方式,确定所述目标软件的哈希值,作为所述目标软件的软件识别信息。

2.根据权利要求1所述的软件识别信息获取方法,其特征在于,在所述获取目标软件的pe文件的dos头之前,还包括:

安装所述目标软件,并基于安装操作更新注册表和系统配置信息;

基于所述注册表或所述系统配置信息,获取所述目标软件的主程序。

3.根据权利要求1所述的软件识别信息获取方法,其特征在于,所述根据所述pe头,在所述主程序中确定指定代码段,包括:

基于所述pe头的首字节,确定所述目标软件的目标运行平台;

按照所述目标运行平台对应的pe头长度确定方式,确定所述pe头的长度;

基于所述pe头的首字节和所述pe头的长度进行偏移,获得所述pe文件的多个节表;

在所述多个节表中确定.text节所在的目标节表,并获取所述目标节表中的所述.text节,作为所述指定代码段。

4.根据权利要求1至3中任一项所述的软件识别信息获取方法,其特征在于,所述从所述指定代码段中提取预定数量的字节,包括:

按照指定提取规则从所述指定代码段中提取预定数量的字节。

5.根据权利要求1至3中任一项所述的软件识别信息获取方法,其特征在于,所述基于所述字节和预定哈希值计算方式,确定所述目标软件的哈希值,包括:

通过md5算法、sha1算法或sm3算法对所述字节进行处理,得到所述目标软件的哈希值。

6.一种软件识别方法,其特征在于,包括:

响应于目标软件的运行请求,获取所述目标软件的软件识别信息;

若所述软件识别信息与软件黑名单内的指定识别信息相匹配,阻止所述目标软件运行,其中,在所述获取所述目标软件的软件识别信息之前,通过执行权利要求1至5中任一项所述方法生成所述软件识别信息。

7.一种软件识别信息获取装置,其特征在于,包括:

第一获取单元,用于获取目标软件的pe文件的dos头;

第二获取单元,用于基于所述dos头的末位字节,获取所述pe文件的pe头;

代码段确定单元,用于根据所述pe头,在所述主程序中确定指定代码段;

字节提取单元,用于从所述指定代码段中提取预定数量的字节;

哈希计算单元,用于基于所述字节和预定哈希值计算方式,确定所述目标软件的哈希值,作为所述目标软件的软件识别信息。

8.一种软件识别装置,其特征在于,包括:

软件识别信息生成单元,用于基于权利要求7所述的软件识别信息获取装置生成软件识别信息;

识别信息获取单元，用于响应于目标软件的运行请求，获取所述目标软件的所述软件识别信息；

软件识别单元，用于若所述软件识别信息与软件黑名单内的指定识别信息相匹配，阻止所述目标软件运行。

9.一种电子设备，其特征在于，包括：至少一个处理器；以及，与所述至少一个处理器通信连接的存储器；

其中，所述存储器存储有可被所述至少一个处理器执行的指令，所述指令被设置为用于执行上述权利要求1至6中任一项所述的方法。

10.一种存储介质，其特征在于，存储有计算机可执行指令，所述计算机可执行指令用于执行如权利要求1至6中任一项所述的方法。

软件识别信息获取方法及装置、电子设备和存储介质

【技术领域】

[0001] 本申请涉及计算机技术领域，尤其涉及一种软件识别信息获取方法及装置、电子设备和存储介质。

【背景技术】

[0002] 为及时阻止违规软件运行，相关技术中为软件设置了软件识别信息，通过软件识别信息可以识别软件，从而进一步判断其是否为被认定为违规的软件。然而，现有的软件识别信息的生成依赖于软件的程序本身，一旦软件更新，软件的程序变动，就需要重新为软件计算新的软件识别信息，而这一计算所消耗的后台维护资源很大。

[0003] 因此，如何减少因软件更新而重新计算软件识别信息所导致的消耗，成为目前亟待解决的技术问题。

【发明内容】

[0004] 本申请实施例提供了一种软件识别信息获取方法及装置、电子设备和存储介质，旨在解决相关技术中因软件更新而重新计算软件识别信息导致后台维护资源很大的技术问题。

[0005] 第一方面，本申请实施例提供了一种软件识别信息获取方法，包括：获取目标软件的pe文件的dos头；基于所述dos头的末位字节，获取所述pe文件的pe头；根据所述pe头，在所述主程序中确定指定代码段；从所述指定代码段中提取预定数量的字节；基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息。

[0006] 在一种可能的设计中，在所述获取目标软件的pe文件的dos头之前，还包括：安装所述目标软件，并基于安装操作更新注册表和系统配置信息；基于所述注册表或所述系统配置信息，获取所述目标软件的主程序。

[0007] 在一种可能的设计中，所述根据所述pe头，在所述主程序中确定指定代码段，包括：基于所述pe头的首字节，确定所述目标软件的目标运行平台；按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度；基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表；在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段。

[0008] 在一种可能的设计中，所述从所述指定代码段中提取预定数量的字节，包括：按照指定提取规则从所述指定代码段中提取预定数量的字节。

[0009] 在一种可能的设计中，所述基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，包括：通过md5算法、sha1算法或sm3算法对所述字节进行处理，得到所述目标软件的哈希值。

[0010] 第二方面，本申请实施例提供了一种软件识别方法，包括：响应于目标软件的运行请求，获取所述目标软件的软件识别信息；若所述软件识别信息与软件黑名单内的指定识别信息相匹配，阻止所述目标软件运行，其中，在所述获取所述目标软件的软件识别信息之

前，通过执行上述第一方面中任一项所述方法生成所述软件识别信息。

[0011] 第三方面，本申请实施例提供了一种软件识别信息获取装置，包括：第一获取单元，用于获取目标软件的pe文件的dos头；第二获取单元，用于基于所述dos头的末位字节，获取所述pe文件的pe头；代码段确定单元，用于根据所述pe头，在所述主程序中确定指定代码段；字节提取单元，用于从所述指定代码段中提取预定数量的字节；哈希计算单元，用于基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息。

[0012] 在一种可能的设计中，所述软件识别信息获取装置还包括：软件安装单元，用于在所述获取目标软件的pe文件的dos头之前，安装所述目标软件，并基于安装操作更新注册表和系统配置信息；主程序获取单元，用于基于所述注册表或所述系统配置信息，获取所述目标软件的主程序。

[0013] 在一种可能的设计中，所述代码段确定单元用于：基于所述pe头的首字节，确定所述目标软件的目标运行平台；按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度；基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表；在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段。

[0014] 在一种可能的设计中，所述字节提取单元用于按照指定提取规则从所述指定代码段中提取预定数量的字节。

[0015] 在一种可能的设计中，所述哈希计算单元用于：通过md5算法、sha1算法或sm3算法对所述字节进行处理，得到所述目标软件的哈希值。

[0016] 第四方面，本申请实施例提供了一种软件识别装置，包括：软件识别信息生成单元，用于基于上述第三方面所述的软件识别信息获取装置生成软件识别信息；识别信息获取单元，用于响应于目标软件的运行请求，获取所述目标软件的所述软件识别信息；软件识别单元，用于若所述软件识别信息与软件黑名单内的指定识别信息相匹配，阻止所述目标软件运行。

[0017] 第五方面，本申请实施例提供了一种电子设备，包括：至少一个处理器；以及，与所述至少一个处理器通信连接的存储器；其中，所述存储器存储有可被所述至少一个处理器执行的指令，所述指令被设置为用于执行上述第一方面所述的方法。

[0018] 第六方面，本申请实施例提供了一种存储介质，存储有计算机可执行指令，所述计算机可执行指令用于执行上述第一方面所述的方法。

[0019] 以上技术方案，可基于指定代码段中预定数量的字节生成软件识别信息，由于所述指定代码段在所述目标软件发生更新时保持不变，则在目标软件发生更新时，基于指定代码段中预定数量的字节生成的软件识别信息也不变。换言之，相对于相关技术中因软件更新而重新计算软件识别信息的技术方案，本申请的技术方案可以在软件更新时保持软件识别信息不变，从而减少因重新生成软件识别信息而消耗的资源，降低维护软件识别信息的成本，提升了软件识别的便利性和安全性。

【附图说明】

[0020] 为了更清楚地说明本申请实施例的技术方案，下面将对实施例中所需要使用的附

图作简单地介绍，显而易见地，下面描述中的附图仅仅是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其它的附图。

[0021] 图1示出了根据本申请的一个实施例的软件识别信息获取方法的流程图；

[0022] 图2示出了根据本申请的另一个实施例的软件识别信息获取方法的流程图；

[0023] 图3示出了根据本申请的一个实施例的软件识别方法的流程图；

[0024] 图4示出了根据本申请的一个实施例的云端与本地端交互过程的示意图；

[0025] 图5示出了根据本申请的一个实施例的软件识别信息获取装置的框图；

[0026] 图6示出了根据本申请的一个实施例的电子设备的框图。

【具体实施方式】

[0027] 下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有作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。

[0028] 图1示出了根据本申请的一个实施例的软件识别信息获取方法的流程图。

[0029] 如图1所示，根据本申请的一个实施例的软件识别信息获取方法包括：

[0030] 步骤102，获取目标软件的pe文件的dos头。

[0031] 步骤104，基于所述dos头的末位字节，获取所述pe文件的pe头。

[0032] pe文件是windows系统中的可执行文件，常见的文件后缀包括但不限于exe、dll、sys、com、ocx等。在32位windows系统上运行的pe文件格式为pe32，在64位windows系统上运行的pe文件格式为pe32+，不同格式的pe文件之差别之一在于其pe头不同。

[0033] pe文件的dos头是一段二进制的数据，其末位字节用于反映pe文件中pe头的位置，故可由此获取pe文件的pe头。

[0034] 步骤106，根据所述pe头，在所述主程序中确定指定代码段。

[0035] pe头相当于指定代码段的位置参照信息，换言之，pe头的位置能够作为确定指定代码段的位置的条件。其中，指定代码段中往往具有一段不会因软件更新而变动的特征段，以不会因软件更新而变动的特征段为基础确定软件的软件识别信息，在软件发生更新时可保证软件的软件识别信息不变，从而避免软件更新所带来的软件识别信息重置，节省计算资源。

[0036] 具体地，可基于所述pe头的首字节，确定所述目标软件的目标运行平台；按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度；基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表；在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段。

[0037] pe头的首字节反映了目标软件的目标运行平台为何，目标运行平台包括但不限于32位平台和64位平台，还可以是任何能够运行目标软件的其他平台。在不同的运行平台中，对pe头长度的计算方式具有差别，因此，可按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度。

[0038] 具体地，在不同的运行平台中可选pe头的大小不一样，在64位平台中，其pe头为IMAGE_OPTIONAL_HEADER64这个结构体，通过sizeof(IMAGE_OPTIONAL_HEADER64)来得到pe

头的长度。

[0039] 而在32位平台中,其pe头为IMAGE_OPTIONAL_HEADER,用sizeof(IMAGE_OPTIONAL_HEADER)得到pe头的长度。

[0040] 在确定pe头的长度后,由于pe头之后分布有多个节表,故可以通过pe头的首字节和pe头的长度进行偏移,来得到多个节表。每个节表用于描述软件所涉及的不同的功能,如存放初始数据,存放调用函数等。同时,每个节表包括多个列表项,或者说多个代码段。进一步地,可遍历各节表,确定.text节所在的节表,将.text节提取出来作为指定代码段。

[0041] 步骤108,从所述指定代码段中提取预定数量的字节。

[0042] 在一种可能的设计中,所述从所述指定代码段中提取预定数量的字节,包括:按照指定提取规则从所述指定代码段中提取预定数量的字节。

[0043] 其中,指定提取规则可为:提取指定代码段中分别位于多个指定位置的多个字节,多个指定位置的数量为预定数量。

[0044] 指定提取规则还可为:提取指定代码段中分别位于多个指定位置的字节,其中,在每个指定位置所提取的字节数量为指定的多个。

[0045] 通过以上技术方案,可使得获取的预定数量的字节更具复杂性,从而提升了以该预定数量的字节所得的软件识别信息的复杂性,提升了软件识别信息对于目标软件的唯一性,提升了软件安全。

[0046] 可选地,预定数量为60。当然,预定数量可以为任何符合对软件识别信息的安全需要的数量,而不限于60这一示例。

[0047] 步骤110,基于所述字节和预定哈希值计算方式,确定所述目标软件的哈希值,作为所述目标软件的软件识别信息。

[0048] 最终,基于指定代码段中预定数量的字节生成软件识别信息,由于该预定数量的字节为指定代码段中不会因软件更新而变动的特征段,则在目标软件发生更新时,基于指定代码段中预定数量的字节生成的软件识别信息也不变。换言之,相对于相关技术中因软件更新而重新计算软件识别信息的技术方案,本申请的技术方案可以在软件更新时保持软件识别信息不变,从而减少因重新生成软件识别信息而消耗的资源,降低维护软件识别信息的成本,提升了软件识别的便利性和安全性。

[0049] 在一种可能的设计中,所述基于所述字节和预定哈希值计算方式,确定所述目标软件的哈希值,包括:通过md5算法、sha1算法或sm3算法对所述字节进行处理,得到所述目标软件的哈希值。

[0050] 其中,md5算法即MD5信息摘要算法,是一种被广泛使用的密码散列函数,可以将从所述指定代码段中提取的预定数量的字节转换为一个128位(16字节)的散列值,作为目标软件的软件识别信息。sha1算法是Hash算法的一种,可将从所述指定代码段中提取的预定数量的字节以512比特的分组为单位处理,输出160比特的消息摘要作为目标软件的软件识别信息。sm3算法是一种密码散列函数标准,将从所述指定代码段中提取的预定数量的字节作为散列函数的输入,经散列函数输出消息摘要作为目标软件的软件识别信息。

[0051] 需要知晓,本申请中的预定哈希值计算方式包括但不限于md5算法、sha1算法或sm3算法,还可以是任何能够把从所述指定代码段中提取的预定数量的字节转换为更为简短的密文的计算方式。

[0052] 图2示出了根据本申请的另一个实施例的软件识别信息获取方法的流程图。

[0053] 如图2所示，根据本申请的另一个实施例的软件识别信息获取方法包括：

[0054] 步骤202，安装目标软件，并基于安装操作更新注册表和系统配置信息。

[0055] 步骤204，基于所述注册表或所述系统配置信息，获取所述目标软件的主程序。

[0056] 若要获取目标软件的软件识别信息，需要从其主程序中获取其pe文件，而只有安装有目标软件，才能够获得其主程序。因此，可安装目标软件，并通过安装后所更新的注册表或所述系统配置信息检索到其主程序。

[0057] 步骤206，获取所述目标软件的pe文件的dos头。

[0058] 步骤208，基于所述dos头的末位字节，获取所述pe文件的pe头。

[0059] pe文件是windows系统中的可执行文件，常见的文件后缀包括但不限于exe、dll、sys、com、ocx等。在32位windows系统上运行的pe文件格式为pe32，在64位windows系统上运行的pe文件格式为pe32+，不同格式的pe文件之差别之一在于其pe头不同。

[0060] pe文件的dos头是一段二进制的数据，其末位字节用于反映pe文件中pe头的位置，故可由此获取pe文件的pe头。

[0061] 步骤210，基于所述pe头的首字节，确定所述目标软件的目标运行平台。

[0062] pe头的首字节反映了目标软件的目标运行平台为何，目标运行平台包括但不限于32位平台和64位平台，还可以是任何能够运行目标软件的其他平台。在不同的运行平台中，对pe头长度的确定方式具有差别，因此，可先确定目标软件所安装于的目标运行平台，以便在后续步骤中针对目标运行平台选择对应的pe头长度的确定方式。

[0063] 步骤212，按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度。

[0064] 在不同的运行平台中可选pe头的大小不一样，在64位平台中，其pe头为IMAGE_OPTIONAL_HEADER64这个结构体，通过sizeof(IMAGE_OPTIONAL_HEADER64)来得到pe头的长度。

[0065] 而在32位平台中，其pe头为IMAGE_OPTIONAL_HEADER，用sizeof(IMAGE_OPTIONAL_HEADER)得到pe头的长度。

[0066] 步骤214，基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表。

[0067] 步骤216，在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段。

[0068] 在确定pe头的长度后，由于pe头之后分布有多个节表，故可以通过pe头的首字节和pe头的长度进行偏移，来得到多个节表。每个节表用于描述软件所涉及的不同的功能，如存放初始数据，存放调用函数等。同时，每个节表包括多个列表项，或者说多个代码段。进一步地，可遍历各节表，确定.text节所在的节表，将.text节提取出来作为指定代码段。

[0069] 步骤218，从所述指定代码段中提取预定数量的字节。

[0070] 可选地，预定数量为60。当然，预定数量可以为任何符合对软件识别信息的安全需要的数量，而不限于60这一示例。

[0071] 步骤220，基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息。

[0072] 最终，基于指定代码段中预定数量的字节生成软件识别信息，由于所述指定代码段在所述目标软件发生更新时保持不变，则在目标软件发生更新时，基于指定代码段中预定数量的字节生成的软件识别信息也不变。

[0073] 通过本技术方案，可以在软件更新时保持软件识别信息不变，从而减少因重新生成软件识别信息而消耗的资源，降低维护软件识别信息的成本，提升了软件识别的便利性和安全性。

[0074] 图3示出了根据本申请的一个实施例的软件识别方法的流程图。

[0075] 如图3所示，根据本申请的一个实施例的软件识别方法包括：

[0076] 步骤302，响应于目标软件的运行请求，获取所述目标软件的软件识别信息。

[0077] 步骤304，若所述软件识别信息与软件黑名单内的指定识别信息相匹配，阻止所述目标软件运行。

[0078] 软件识别信息可用于识别目标软件的身份，通过将目标软件的软件识别信息与软件黑名单内的指定识别信息进行匹配，可判断目标软件是否为软件黑名单的一员，从而在目标软件为软件黑名单的一员的情况下，阻止所述目标软件运行，保护系统安全和网络安全。

[0079] 其中，在步骤302之前，还包括：通过执行上述任一实施例中任一项所述的技术方案来生成软件识别信息。因此，本技术方案具有上述全部技术效果，在此处不再赘述。

[0080] 至此可以知晓，所述软件识别信息的生成过程可以在云端进行，也可以在本地端进行。

[0081] 图4示出了根据本申请的一个实施例的云端与本地端交互过程的示意图。如图4所示，在云端设置有自动化下载安装模块、分析主程序模块、哈希提取模块和特征哈希数据库，而在本地端则设置有外部调用模块。

[0082] 具体地，在云端，自动化下载安装模块安装所述目标软件，并基于安装操作更新注册表和系统配置信息，以便基于所述注册表或所述系统配置信息，获取所述目标软件的主程序。

[0083] 分析主程序模块则获取目标软件的pe文件的dos头，并根据所述dos头的末位字节，获取所述pe文件的pe头，接着，基于所述pe头的首字节，确定所述目标软件的目标运行平台，然后，按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度，接下来，基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表，至此，在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段。

[0084] 接下来，哈希提取模块从所述指定代码段中提取预定数量的字节，并基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息。

[0085] 特征哈希数据库则可将哈希提取模块生成的目标软件的软件识别信息进行存储。

[0086] 在本地端，响应于目标软件的运行请求，通过外部调用模块访问云端的特征哈希数据库，调用目标软件的软件识别信息。若所述软件识别信息与软件黑名单内的指定识别信息相匹配，本地端阻止所述目标软件运行，否则，本地端允许所述目标软件运行。

[0087] 本技术方案中，软件识别信息可用于识别目标软件的身份，通过将目标软件的软

件识别信息与软件黑名单内的指定识别信息进行匹配，可判断目标软件是否为软件黑名单的一员，从而在目标软件为软件黑名单的一员的情况下，阻止所述目标软件运行，保护系统安全和网络安全。

[0088] 图5示出了根据本申请的一个实施例的软件识别信息获取装置的框图。

[0089] 如图5所示，根据本申请的一个实施例的软件识别信息获取装置500包括：第一获取单元502，用于获取目标软件的pe文件的dos头；第二获取单元504，用于基于所述dos头的末位字节，获取所述pe文件的pe头；代码段确定单元506，用于根据所述pe头，在所述主程序中确定指定代码段；字节提取单元508，用于从所述指定代码段中提取预定数量的字节；哈希计算单元510，用于基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息。

[0090] 在一种可能的设计中，所述软件识别信息获取装置500还包括：软件安装单元，用于在所述获取目标软件的pe文件的dos头之前，安装所述目标软件，并基于安装操作更新注册表和系统配置信息；主程序获取单元，用于基于所述注册表或所述系统配置信息，获取所述目标软件的主程序。

[0091] 在一种可能的设计中，所述代码段确定单元506用于：基于所述pe头的首字节，确定所述目标软件的目标运行平台；按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度；基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表；在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段。

[0092] 在一种可能的设计中，所述字节提取单元508用于按照指定提取规则从所述指定代码段中提取预定数量的字节。

[0093] 在一种可能的设计中，所述哈希计算单元510用于：通过md5算法、sha1算法或sm3算法对所述字节进行处理，得到所述目标软件的哈希值。

[0094] 该软件识别信息获取装置500使用上述实施例中任一项所述的方案，因此，具有上述所有技术效果，在此不再赘述。

[0095] 另外，本申请实施例还提供了一种软件识别装置，包括：软件识别信息生成单元，用于基于软件识别信息获取装置500生成软件识别信息；识别信息获取单元，用于响应于目标软件的运行请求，获取所述目标软件的所述软件识别信息；软件识别单元，用于若所述软件识别信息与软件黑名单内的指定识别信息相匹配，阻止所述目标软件运行。

[0096] 该软件识别装置使用软件识别信息获取装置500的所有技术效果，在此不再赘述。

[0097] 图6示出了根据本申请的一个实施例的电子设备的框图。

[0098] 如图6所示，本申请的一个实施例的电子设备600，包括至少一个存储器602；以及，与所述至少一个存储器602通信连接的处理器604；其中，所述存储器存储有可被所述至少一个处理器604执行的指令，所述指令被设置为用于执行上述任一实施例中所述的方案。因此，该电子设备600具有和上述任一实施例中相同的技术效果，在此不再赘述。

[0099] 本申请实施例的电子设备以多种形式存在，包括但不限于：

[0100] (1)移动通信设备：这类设备的特点是具备移动通信功能，并且以提供话音、数据通信为主要目标。这类终端包括：智能手机、多媒体手机、功能性手机，以及低端手机等。

[0101] (2)超移动个人计算机设备：这类设备属于个人计算机的范畴，有计算和处理功

能,一般也具备移动上网特性。这类终端包括:PDA、MID和UMPC设备等。

[0102] (3)便携式娱乐设备:这类设备可以显示和播放多媒体内容。该类设备包括:音频、视频播放器,掌上游戏机,电子书,以及智能玩具和便携式车载导航设备。

[0103] (4)服务器:提供计算服务的设备,服务器的构成包括处理器、硬盘、内存、系统总线等,服务器和通用的计算机架构类似,但是由于需要提供高可靠的服务,因此在处理能力、稳定性、可靠性、安全性、可扩展性、可管理性等方面要求较高。

[0104] (5)其他具有数据交互功能的电子装置。

[0105] 需要知晓,本申请建立在对windows系统下的pe文件的特征提取的基础之上,因此,可适用于各类Surface设备。

[0106] 另外,本申请实施例提供了一种存储介质,存储有计算机可执行指令,所述计算机可执行指令用于执行以下步骤:获取目标软件的pe文件的dos头;基于所述dos头的末位字节,获取所述pe文件的pe头;根据所述pe头,在所述主程序中确定指定代码段;从所述指定代码段中提取预定数量的字节;基于所述字节和预定哈希值计算方式,确定所述目标软件的哈希值,作为所述目标软件的软件识别信息。

[0107] 需要说明的是,上述关于存储介质或电子设备所能实现的功能或步骤,可对应参阅前述方法实施例中的相关描述,为避免重复,这里不再一一描述。

[0108] 以上结合附图详细说明了本申请的技术方案,通过本申请的技术方案,可以在软件更新时保持软件识别信息不变,从而减少因重新生成软件识别信息而消耗的资源,降低维护软件识别信息的成本,提升了软件识别的便利性和安全性。

[0109] 取决于语境,如在此所使用的词语“如果”可以被解释成为“在……时”或“当……时”或“响应于确定”或“响应于检测”。类似地,取决于语境,短语“如果确定”或“如果检测(陈述的条件或事件)”可以被解释成为“当确定时”或“响应于确定”或“当检测(陈述的条件或事件)时”或“响应于检测(陈述的条件或事件)”。

[0110] 在本申请实施例中使用的术语是仅仅出于描述特定实施例的目的,而非旨在限制本申请。在本申请实施例和所附权利要求书中所使用的单数形式的“一种”、“所述”和“该”也旨在包括多数形式,除非上下文清楚地表示其他含义。

[0111] 在本申请所提供的几个实施例中,应该理解到,所揭露的系统、装置和方法,可以通过其它的方式实现。例如,以上所描述的装置实施例仅仅是示意性的,例如,所述单元的划分,仅仅为一种逻辑功能划分,实际实现时可以有另外的划分方式,例如,多个单元或组件可以结合或者可以集成到另一个系统,或一些特征可以忽略,或不执行。另一点,所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口,装置或单元的间接耦合或通信连接,可以是电性,机械或其它的形式。

[0112] 另外,在本申请各个实施例中的各功能单元可以集成在一个处理单元中,也可以是各个单元单独物理存在,也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现,也可以采用硬件加软件功能单元的形式实现。

[0113] 本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程,是可以通过计算机程序来指令相关的硬件来完成,所述的计算机程序可存储于一非易失性计算机可读取存储介质中,该计算机程序在执行时,可包括如上述各方法的实施例的流程。其中,本申请所提供的各实施例中所使用的对存储器、存储、数据库或其它介质的任何引用,均可

包括非易失性和/或易失性存储器。非易失性存储器可包括只读存储器(ROM)、可编程ROM(PROM)、电可编程ROM(EPROM)、电可擦除可编程ROM(EEPROM)或闪存。易失性存储器可包括随机存取存储器(RAM)或者外部高速缓冲存储器。作为说明而非局限,RAM以多种形式可得,诸如静态RAM(SRAM)、动态RAM(DRAM)、同步DRAM(SDRAM)、双数据率SDRAM(DDRSDRAM)、增强型SDRAM(ESDRAM)、同步链路(Synchlink)DRAM(SLDRAM)、存储器总线(Rambus)直接RAM(RDRAM)、直接存储器总线动态RAM(DRDRAM)、以及存储器总线动态RAM(RDRAM)等。

[0114] 以上所述实施例仅用以说明本发明的技术方案,而非对其限制;尽管参照前述实施例对本发明进行了详细的说明,本领域的普通技术人员应当理解:其依然可以对前述各实施例所记载的技术方案进行修改,或者对其中部分技术特征进行等同替换;而这些修改或者替换,并不使相应技术方案的本质脱离本发明各实施例技术方案的精神和范围,均应包含在本发明的保护范围之内。

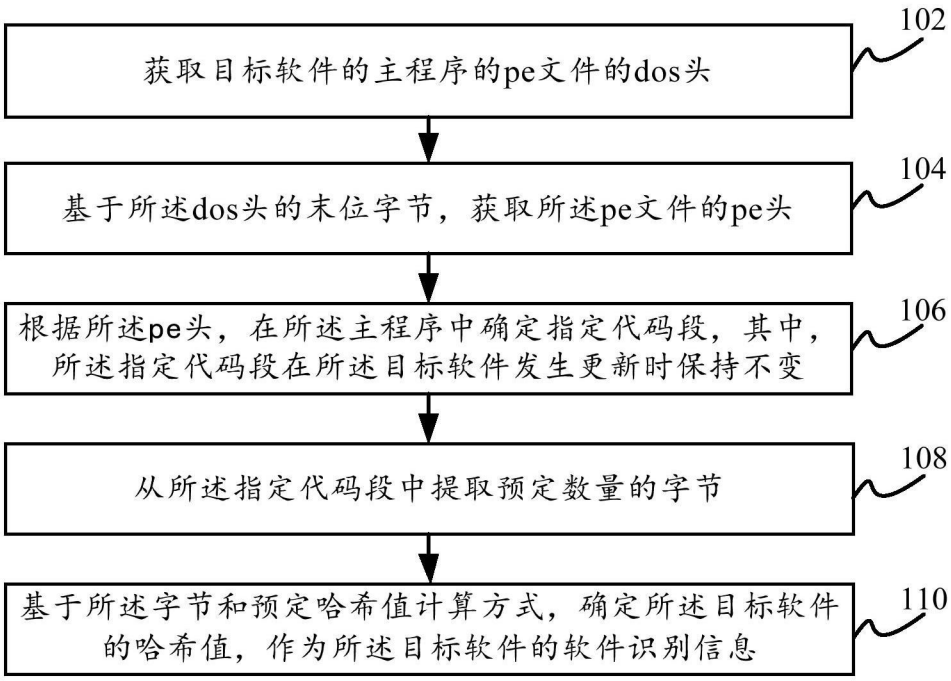

102
获取目标软件的主程序的pe文件的dos头
104
基于所述dos头的末位字节，获取所述pe文件的pe头
106
根据所述pe头，在所述主程序中确定指定代码段，其中，所述指定代码段在所述目标软件发生更新时保持不变
108
从所述指定代码段中提取预定数量的字节
110
基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息

图1

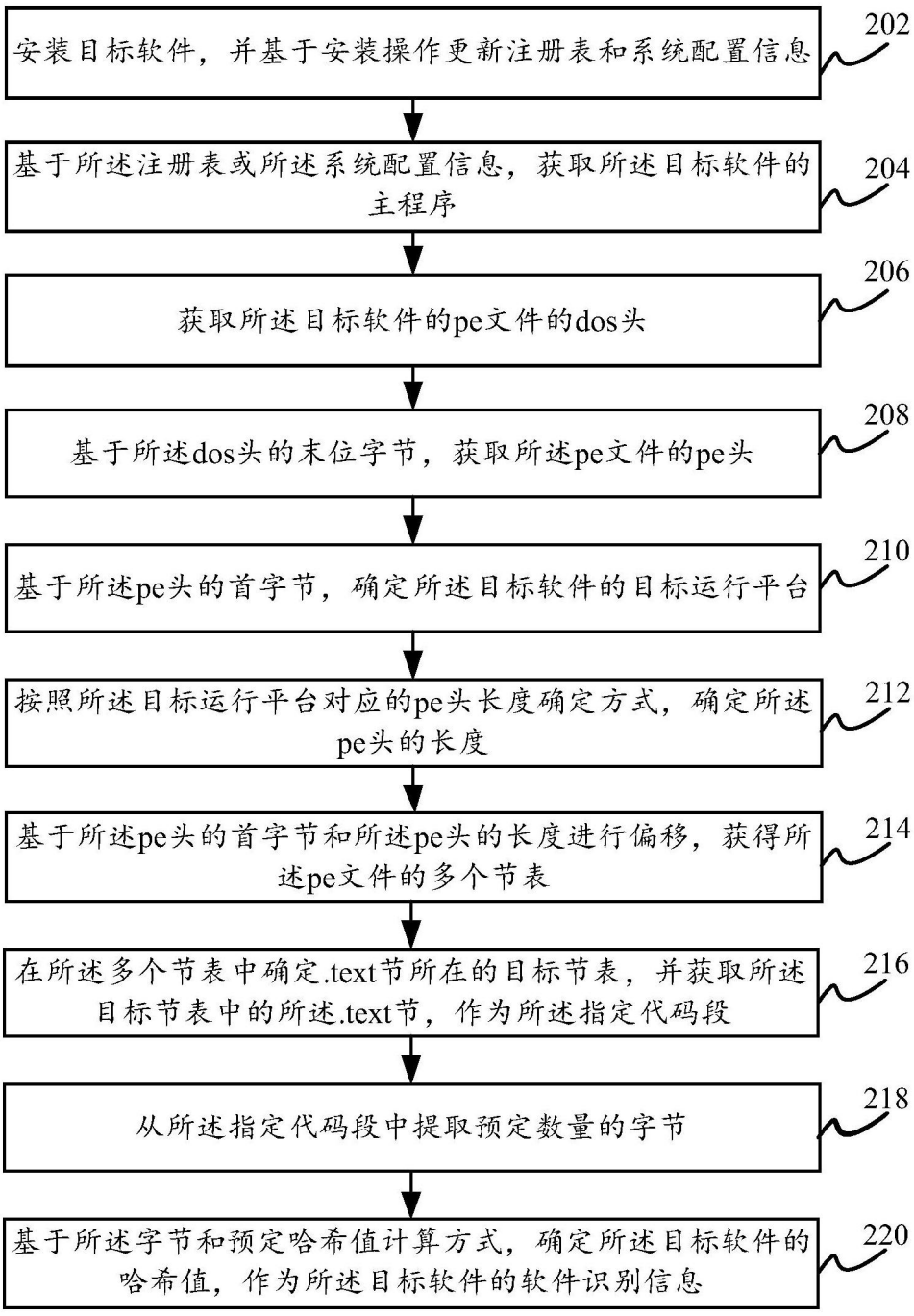

安装目标软件，并基于安装操作更新注册表和系统配置信息
202
基于所述注册表或所述系统配置信息，获取所述目标软件的主程序
204
获取所述目标软件的pe文件的dos头
206
基于所述dos头的末位字节，获取所述pe文件的pe头
208
基于所述pe头的首字节，确定所述目标软件的目标运行平台
210
按照所述目标运行平台对应的pe头长度确定方式，确定所述pe头的长度
212
基于所述pe头的首字节和所述pe头的长度进行偏移，获得所述pe文件的多个节表
214
在所述多个节表中确定.text节所在的目标节表，并获取所述目标节表中的所述.text节，作为所述指定代码段
216
从所述指定代码段中提取预定数量的字节
218
基于所述字节和预定哈希值计算方式，确定所述目标软件的哈希值，作为所述目标软件的软件识别信息
220

图2

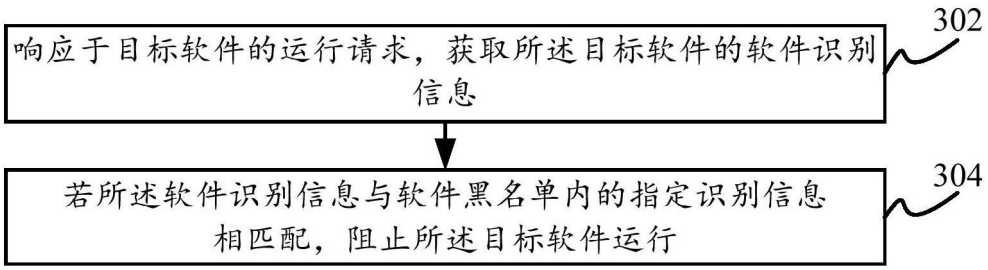

响应于目标软件的运行请求，获取所述目标软件的软件识别信息
302
若所述软件识别信息与软件黑名单内的指定识别信息相匹配，阻止所述目标软件运行
304

图3

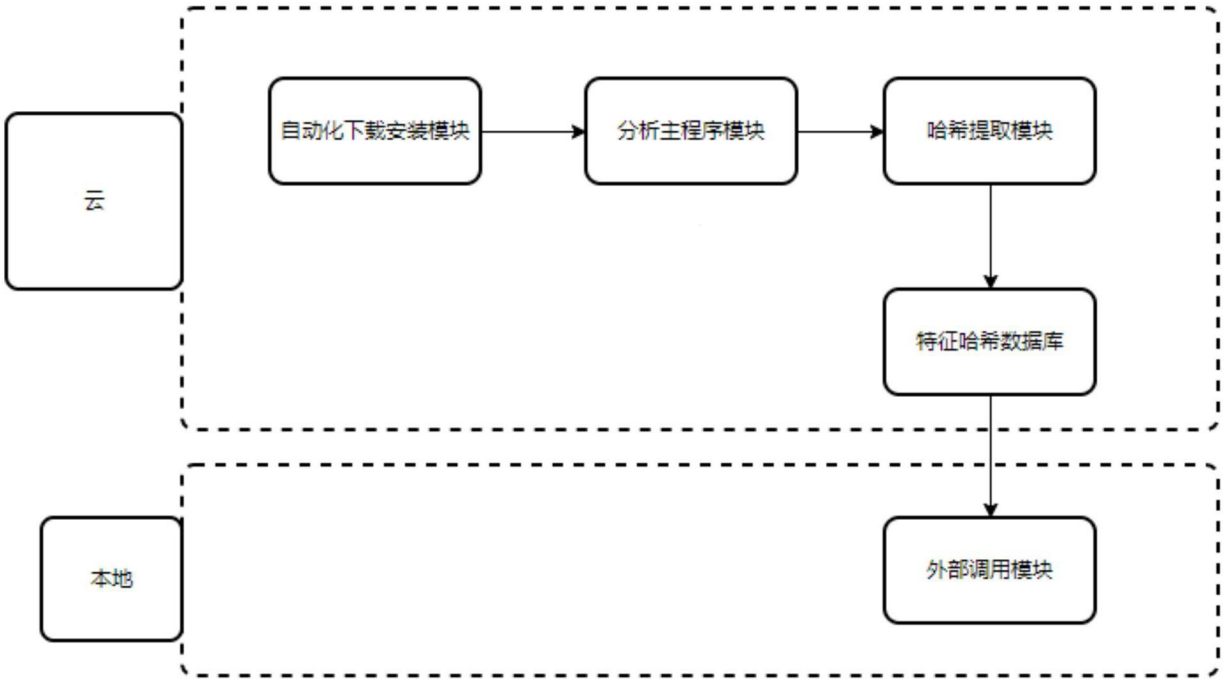

云
自动化下载安装模块
分析主程序模块
哈希提取模块
特征哈希数据库
本地
外部调用模块

图4


第一获取单元502
第二获取单元504
代码段确定单元506
字节提取单元508
哈希计算单元510
软件识别信息获取装置500

图5

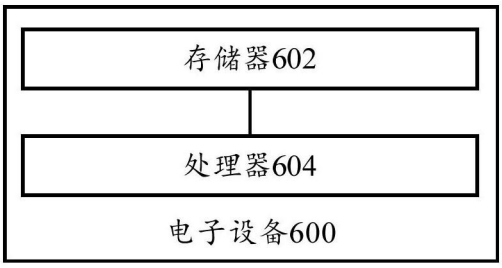

存储器602
处理器604
电子设备600

图6