



# (12) 发明专利

(10) 授权公告号 CN 111475234 B

(45) 授权公告日 2023.01.10

(21) 申请号 202010279076.8

(56) 对比文件

(22) 申请日 2020.04.10

CN 110932822 A, 2020.03.27

(65) 同一申请的已公布的文献号

审查员 邹孝杰

申请公布号 CN 111475234 A

(43) 申请公布日 2020.07.31

(73) 专利权人 苏州浪潮智能科技有限公司

地址 215100 江苏省苏州市吴中区吴中经

济开发区郭巷街道官浦路1号9幢

(72) 发明人 张一罡

(74) 专利代理机构 北京集佳知识产权代理有限

公司 11227

专利代理师 田媛媛

(51) Int. Cl.

G06F 9/448 (2018.01)

G06F 40/126 (2020.01)

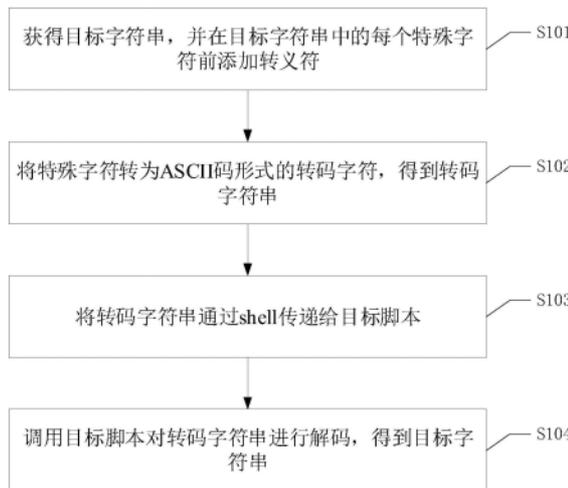
权利要求书2页 说明书7页 附图2页

## (54) 发明名称

一种字符串传递方法、装置、计算机及可读存储介质

## (57) 摘要

本发明公开了一种字符串传递方法、装置、计算机及可读存储介质,该方法包括以下步骤:获得目标字符串,并在目标字符串中的每个特殊字符前添加转义符;特殊字符为无法准确传递至目标脚本的字符;将特殊字符转为ASCII码形式的转码字符,得到转码字符串;将转码字符串通过shell传递给目标脚本;调用目标脚本对转码字符串进行解码,得到目标字符串。相较于现有的复杂转义,本方法实现更为简单,能够有效避免特殊字符被shell特殊处理。



1. 一种字符串传递方法,其特征在于,包括:

获得目标字符串,并在所述目标字符串中的每个特殊字符前添加转义符;所述特殊字符为无法准确传递至目标脚本的字符;

将所述特殊字符转为ASCII码形式的转码字符,得到转码字符串;

将所述转码字符串通过shell传递给所述目标脚本;

调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串;

其中,将所述特殊字符转为ASCII码形式,得到转码字符串,包括:

将所述特殊字符串转为x+ASCII码形式,得到所述转码字符串;其中,所述x+ASCII码为x+16进制的ASCII数字;

调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串,包括:

将所述转码字符串中的每一个x替换为\\x;

利用字符串显示命令对所述转码字符串中的转码字符进行解码并更替,得到所述目标字符串;所述字符串显示命令为echo -e命令;以便使用所述echo -e命令来解码\ASCII字符,原理为-e选项会对特殊字符\x进行特殊处理,从而使\x发挥了16进制转义的作用。

2. 根据权利要求1所述的字符串传递方法,其特征在于,将所述特殊字符串转为x+ASCII码形式,得到所述转码字符串,包括:

利用字符串格式化命令,将所述特殊字符串转为x+ASCII码形式,得到所述转码字符串。

3. 根据权利要求1所述的字符串传递方法,其特征在于,所述获得目标字符串,包括:

判断待传递字符串是否包括所述特殊字符;

如果是,则将所述待传递字符串作为所述目标字符串。

4. 根据权利要求1所述的字符串传递方法,其特征在于,在调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串之后,还包括:

利用所述目标字符串对目标文件进行修改。

5. 根据权利要求4所述的字符串传递方法,其特征在于,利用所述目标字符串对所述目标文件进行修改,包括:

调用写文件工具,利用所述目标字符串对所述目标文件进行修改。

6. 一种字符串传递装置,其特征在于,包括:

Linux程序,用于获得目标字符串,并在所述目标字符串中的每个特殊字符前添加转义符;所述特殊字符为无法准确传递至目标脚本的字符;将所述特殊字符转为ASCII码形式的转码字符,得到转码字符串;将所述转码字符串通过shell传递给所述目标脚本;

目标脚本,用于接受所述Linux程序的调用,对所述转码字符串进行解码,得到所述目标字符串;

其中,将所述特殊字符转为ASCII码形式,得到转码字符串,包括:

将所述特殊字符串转为x+ASCII码形式,得到所述转码字符串;其中,所述x+ASCII码为x+16进制的ASCII数字;

调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串,包括:

将所述转码字符串中的每一个x替换为\\x;

利用字符串显示命令对所述转码字符串中的转码字符进行解码并更替,得到所述目标

字符串;所述字符串显示命令为echo -e命令;以便使用所述echo -e命令来解码\ASCII字符,原理为-e选项会对特殊字符\x进行特殊处理,从而使\x发挥了16进制转义的作用。

7.一种计算机,其特征在于,包括:

存储器,用于存储计算机程序;

处理器,用于执行所述计算机程序时实现如权利要求1至5任一项所述字符串传递方法的步骤。

8.一种可读存储介质,其特征在于,所述可读存储介质上存储有计算机程序,所述计算机程序被处理器执行时实现如权利要求1至5任一项所述字符串传递方法的步骤。

## 一种字符串传递方法、装置、计算机及可读存储介质

### 技术领域

[0001] 本发明涉及计算机软件技术领域,特别是涉及一种字符串传递方法、装置、计算机及可读存储介质。

### 背景技术

[0002] 在Linux程序设计中,经常会涉及在程序中调用脚本的场景,在大部分情况下,程序传递给脚本的参数都是比较简单的。但在某些情况下,比如需要将含有多种特殊字符的密码写入文件,那么就涉及到特殊字符的传递。如果在Linux环境中直接将含有特殊字符的字符串作为参数传递给脚本,那么shell会将某些特殊字符预先进行特殊处理。比如“\$\$”将被作为进程的PID(Process Identification,操作系统里的进程识别号,即进程标识符),也就是说,如果参数字符串含有“\$\$”,那么脚本接收到的将会是一串表示进程PID的数字字符,这就会导致脚本接收一个错误的参数,进而导致执行出错。

[0003] 针对于这种情况,就需要对这些含有特殊字符的字符串进行处理。其中一种比较经典的方法是使用字符转义符“\”对特殊字符进行转义。比如\$\$可以加转义符变成“\\$\\$”。当特殊字符比较少时,这种方式是可行的,但是当特殊字符数量很多时,或者含有特殊字符的参数很多时,逐个加转义符则会很麻烦。而且在某些情况下,脚本获得这些字符串后,需要将它们写入文件,这样就要对特殊字符进行两层转义,一层用于程序和脚本间参数的传递,一层用于脚本和文件参数的传递。转义的层数过多,就会使得转义过程非常麻烦,容易出现错误。

[0004] 综上所述,如何有效地解决特殊字符的传递等问题,是目前本领域技术人员急需解决的技术问题。

### 发明内容

[0005] 本发明的目的是提供一种字符串传递方法、装置、计算机及可读存储介质,以Linux程序调用目标脚本时,能够准确无误的将特殊字符传递给目标脚本。

[0006] 为解决上述技术问题,本发明提供如下技术方案:

[0007] 一种字符串传递方法,其特征在于,包括:

[0008] 获得目标字符串,并在所述目标字符串中的每个特殊字符前添加转义符;所述特殊字符为无法准确传递至目标脚本的字符;

[0009] 将所述特殊字符转为ASCII码形式的转码字符,得到转码字符串;

[0010] 将所述转码字符串通过shell传递给所述目标脚本;

[0011] 调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串。

[0012] 优选地,将所述特殊字符转为ASCII码形式,得到转码字符串,包括:

[0013] 将所述特殊字符串转为x+ASCII码形式,得到所述转码字符串。

[0014] 优选地,将所述特殊字符串转为x+ASCII码形式,得到所述转码字符串,包括:

[0015] 利用字符串格式化命令,将所述特殊字符串转为x+ASCII码形式,得到所述转码字

字符串。

[0016] 优选地,调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串,包括:

[0017] 将所述转码字符串中的每一个x替换为\\x;

[0018] 利用字符串显示命令对所述转码字符串中的转码字符进行解码并更替,得到所述目标字符串。

[0019] 优选地,所述获得目标字符串,包括:

[0020] 判断待传递字符串是否包括所述特殊字符;

[0021] 如果是,则将所述待传递字符串作为所述目标字符串。

[0022] 优选地,在调用所述目标脚本对所述转码字符串进行解码,得到所述目标字符串之后,还包括:

[0023] 利用所述目标字符串对目标文件进行修改。

[0024] 优选地,利用所述目标字符串对所述目标文件进行修改,包括:

[0025] 调用写文件工具,利用所述目标字符串对所述目标文件进行修改。

[0026] 一种字符串传递装置,包括:

[0027] Linux程序,用于获得目标字符串,并在所述目标字符串中的每个特殊字符前添加转义符;所述特殊字符为无法准确传递至目标脚本的字符;将所述特殊字符转为ASCII码形式的转码字符,得到转码字符串;将所述转码字符串通过shell传递给所述目标脚本;

[0028] 目标脚本,用于接受所述Linux程序的调用,对所述转码字符串进行解码,得到所述目标字符串。

[0029] 一种计算机,包括:

[0030] 存储器,用于存储计算机程序;

[0031] 处理器,用于执行所述计算机程序时实现上述字符串传递方法的步骤。

[0032] 一种可读存储介质,所述可读存储介质上存储有计算机程序,所述计算机程序被处理器执行时实现上述字符串传递方法的步骤。

[0033] 应用本发明实施例所提供的方法,获得目标字符串,并在目标字符串中的每个特殊字符前添加转义符;特殊字符为无法准确传递至目标脚本的字符;将特殊字符转为ASCII码形式的转码字符,得到转码字符串;将转码字符串通过shell传递给目标脚本;调用目标脚本对转码字符串进行解码,得到目标字符串。

[0034] 在本方法中,当待传递给目标脚本的字符串中包括特殊字符串时,首先在每个特殊字符串前添加转义符,然后将特殊字符转为ASCII码形式的转码字符,得到转码字符串。即,转码字符串中不再包括特殊字符,可顺利通过shell传递给目标脚本。然后调用目标脚本时,目标脚本对转码字符串进行解码,即可得到目标字符串。也就是说,在传递目标字符串之前进行转义,转码,便可得到能够传递给目标脚本的转码字符串,而在传递转码字符串之后,调用目标脚本对转码字符串进行解码,即可得到目标字符串。如此,便可使得原本无法准确传递至目标脚本的特殊字符,准确地传递给了目标脚本。相较于现有的复杂转义,本方法实现更为简单,能够有效避免特殊字符被shell特殊处理。

[0035] 相应地,本发明实施例还提供了与上述字符串传递方法相对应的字符串传递装置、计算机和可读存储介质,具有上述技术效果,在此不再赘述。

## 附图说明

[0036] 为了更清楚地说明本发明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,下面描述中的附图仅仅是本发明的一些实施例,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他的附图。

[0037] 图1为本发明实施例中一种字符串传递方法的实施流程图;

[0038] 图2为本发明实施例中一种字符串传递装置的结构示意图;

[0039] 图3为本发明实施例中一种计算机的结构示意图;

[0040] 图4为本发明实施例中一种计算机的具体结构示意图。

## 具体实施方式

[0041] 为了使本技术领域的人员更好地理解本发明方案,下面结合附图和具体实施方式对本发明作进一步的详细说明。显然,所描述的实施例仅仅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。

[0042] 实施例一:

[0043] 请参考图1,图1为本发明实施例中一种字符串传递方法的流程图,该方法可应用于安装了linux系统的计算机中,有该计算机中的程序实现该方法的步骤。该方法包括以下步骤:

[0044] S101、获得目标字符串,并在目标字符串中的每个特殊字符前添加转义符。

[0045] 其中,特殊字符为无法准确传递至目标脚本的字符。例如,特殊字符可包括但不限于“\$\$”、“<”、“>”、“;”、“”、“”、“”、“|”、“!”、“(”、“&”。其中,“\$\$”,会被shell被作为进程的PID,shell会进行特殊处理,那么脚本接收到的将会是一串表示进程PID的数字字符,即导致脚本接收一个错误的参数,进而导致执行出错;对于“<”、“>”,linux系统误以为是输出导向符,“<”、“>”不会被进行传递,且linux系统会报找不到文件错误;“;”字符,会被linux系统当成命令结束符,linux系统把“;”字符后面的内容当成命令来执行;单个的单引号“'”,双引号“””,倒引号“`”会被linux系统认为数据没有输出完,造成系统等待输入的情况,“|”会被linux系统当成管道符号,“|”字符后面的内容会被当成命令来执行;“!”字符会被当成逻辑“非”,linux系统会报错“event not found”,“(”字符会被linux系统认为是指令群组而报语法错误;“&”会被linux系统认为是后台执行指令,这会导致“&”以及“&”后面的内容不会被传递。当这些特殊字符组合起来使用时,甚至可能得到意想不到的结果。

[0046] 其中,目标脚本即可需接收目标字符串的脚本,该脚本本身可为文件修改脚本,也可为其他脚本。

[0047] 在本实施例中,可预先将无法准确传递至目标脚本的字符进行存储,以便对需要传递至脚本的字符串进行比对,确定是否需要进行转义处理。也就是说,获得目标字符串,包括:

[0048] 步骤一、判断待传递字符串是否包括特殊字符;

[0049] 步骤二、如果是,则将待传递字符串作为目标字符串。

[0050] 其中,待传递字符串可具体为用户输入的密码或配置信息等需要传递给目标脚本

的字符串。

[0051] 可通过遍历查询待传递字符串的方式,确定出待传递字符串中是否包括特殊字符,如果包括特殊字符串,则将该待传递字符串作为目标字符串,如果不包括特殊字符串则可忽略,即可直接传递给目标脚本。

[0052] 获得目标字符串之后,则可对目标字符串中的特殊字符进行标注,以便进行转义处理。具体的,可在每一个特殊字符前添加转义符,即“/”。例如,若目标字符串中包括特殊字符“>”,则添加转义符,即“/>”;若目标字符串为“12eD>”则添加了转义符之后,得到的转码字符串为“12eD/>”。当然,在实际应用中,特殊字符的数量和种类可为多个,在此不再一一列举。

[0053] S102、将特殊字符转为ASCII码形式的转码字符,得到转码字符串。

[0054] ASCII(American Standard Code for Information Interchange),即基于拉丁字母的一套电脑编码系统,是一种通用的信息交换标准。

[0055] 在本实施例中,将添加了转义符的特殊字符转为ASCII码形式的转码字符,然后便可得到转码字符串。即转码字符串中无特殊字符,取而代之的是特殊字符对应的ASCII码形式的转码字符串。采用ASCII码格式不但能解决特殊字符的问题,还能在目标脚本接收到转码字符串后被快速解码。

[0056] 优选地,考虑到传递至目标脚本之后,还需调用目标脚本对转码字符串进行解码处理,为了提高解码效率,在本实施例中还可将特殊字符转为x+ASCII码形式。也就是说,可将特殊字符串转为x+ASCII码形式,得到转码字符串。将特殊字符转化为x+ASCII码的形式的方式有多种。进一步地,为了加快转码效率,可利用字符串格式化命令,将特殊字符串转为x+ASCII码形式,得到转码字符串。即,遍历目标字符串找出添加了转义符的特殊字符,对每个特殊字符,转化成x+ASCII码的形式。可使用sprintf()函数(即字符串格式化命令)指定16进制格式输出,能够快速达到转码的效果。其中,x主要是为了方便脚本对ASCII字符串进行16进制转义,从而起到解码ASCII码的作用。这一步最终会形成一个新的由字符x和ASCII码组成的字符串,其形式为xhhxhhxhhxhh...,其中hh为16进制ASCII数字。

[0057] 对目标字符串中的特殊字符串进行转码之后,便可获得转码字符串。

[0058] S103、将转码字符串通过shell传递给目标脚本。

[0059] 由于转码字符串中不再包括特殊字符,因而shell不会对特殊字符进行特殊处理,如不会在更改转码字符串中的字符。

[0060] S104、调用目标脚本对转码字符串进行解码,得到目标字符串。

[0061] 解码的过程与转码、转义相对应。即,对转码字符进行解码,得到特殊字符,并基于特殊字符串将转码字符串中的转码字符更换为特殊字符,得到目标字符串。若转码字符串中的转码字符为ASCII码形式,则直接将ASCII码形式的转码字符解码,得到特殊字符。

[0062] 若转码字符串中的转码字符为x+ASCII码形式,则解码的具体实现过程,可包括:

[0063] 步骤一、将转码字符串中的每一个x替换为\\x;

[0064] 步骤二、利用字符串显示命令对转码字符串中的转码字符进行解码并更替,得到目标字符串。

[0065] 即,目标脚本在接收到转码字符串后,需要有两步操作来解码ASCII字符串。第一步可使用sed命令将字符串里每一个“x”替换为为“\\x”。其中第一个“\”是为了对第二个

“\”进行转义,而第二个“\”和“x”结合起来形成的“\x”表示16进制转义字符。第二步,可使用echo-e命令(即字符串显示命令)来解码ASCII字符,主要原理是-e选项会对特殊字符“\x”进行特殊处理,从而使“\x”发挥了16进制转义的作用。经过这一步,ASCII数字已被转化为普通格式的字符,即恢复为特殊字符的样式。

[0066] 应用本发明实施例所提供的方法,获得目标字符串,并在目标字符串中的每个特殊字符前添加转义符;特殊字符为无法准确传递至目标脚本的字符;将特殊字符转为ASCII码形式的转码字符,得到转码字符串;将转码字符串通过shell传递给目标脚本;调用目标脚本对转码字符串进行解码,得到目标字符串。

[0067] 在本方法中,当待传递给目标脚本的字符串中包括特殊字符串时,首先在每个特殊字符串前添加转义符,然后将特殊字符转为ASCII码形式的转码字符,得到转码字符串。即,转码字符串中不再包括特殊字符,可顺利通过shell传递给目标脚本。然后调用目标脚本时,目标脚本对转码字符串进行解码,即可得到目标字符串。也就是说,在传递目标字符串之前进行转义,转码,便可得到能够传递给目标脚本的转码字符串,而在传递转码字符串之后,调用目标脚本对转码字符串进行解码,即可得到目标字符串。如此,便可使得原本无法准确传递至目标脚本的特殊字符,准确地传递给了目标脚本。相较于现有的复杂转义,本方法实现更为简单,能够有效避免特殊字符被shell特殊处理。

[0068] 需要说明的是,基于上述实施例,本发明实施例还提供了相应的改进方案。在优选/改进实施例中涉及与上述实施例中相同步骤或相应步骤之间可相互参考,相应的有益效果也可相互参照,在本文的优选/改进实施例中不再一一赘述。

[0069] 优选地,在执行上述步骤S104之后,目标脚本即得到了目标字符串,此时可直接利用目标字符串对目标文件进行修改。具体的,可调用写文件工具,利用目标字符串对目标文件进行修改。

[0070] 转码字符串中的转码字符被转义为普通字符后,就可以进行写文件的操作了。例如,可采用常用的写文件的工具有awk,sed。在使用这些工具进行修改文件的操作时,传入的字符串参数里某些特殊字符也会被处理,比如“\”会被作为转义符处理,这样“\n”会首先被认为是“n”,而达不到空格的作用。但是,由于步骤S101中,字符串转义操作中,已经对特殊字符加了转义,也就是字符串里的“\n”已经在第一步中被处理为“\\n”,所以即使使用awk或者sed处理执行了步骤S104后得到的目标字符串,也不会发生错误。

[0071] 实施例二:

[0072] 相应于上面的方法实施例,本发明实施例还提供了一种字符串传递装置,下文描述的字符串传递装置与上文描述的字符串传递方法可相互对应参照。

[0073] 参见图2所示,该装置包括以下模块:

[0074] Linux程序100,用于获得目标字符串,并在目标字符串中的每个特殊字符前添加转义符;特殊字符为无法准确传递至目标脚本的字符;将特殊字符转为ASCII码形式的转码字符,得到转码字符串;将转码字符串通过shell传递给目标脚本;

[0075] 目标脚本200,用于接受Linux程序的调用,对转码字符串进行解码,得到目标字符串。

[0076] 应用本发明实施例所提供的装置,获得目标字符串,并在目标字符串中的每个特殊字符前添加转义符;特殊字符为无法准确传递至目标脚本的字符;将特殊字符转为ASCII

码形式的转码字符,得到转码字符串;将转码字符串通过shell传递给目标脚本;调用目标脚本对转码字符串进行解码,得到目标字符串。

[0077] 在本装置中,当待传递给目标脚本的字符串中包括特殊字符串时,首先在每个特殊字符串前添加转义符,然后将特殊字符转为ASCII码形式的转码字符,得到转码字符串。即,转码字符串中不再包括特殊字符,可顺利通过shell传递给目标脚本。然后调用目标脚本时,目标脚本对转码字符串进行解码,即可得到目标字符串。也就是说,在传递目标字符串之前进行转义,转码,便可得到能够传递给目标脚本的转码字符串,而在传递转码字符串之后,调用目标脚本对转码字符串进行解码,即可得到目标字符串。如此,便可使得原本无法准确传递至目标脚本的特殊字符,准确地传递给了目标脚本。相较于现有的复杂转义,本装置实现更为简单,能够有效避免特殊字符被shell特殊处理。

[0078] 在本发明的一种具体实施方式中,Linux程序100,具体用于将特殊字符串转为x+ASCII码形式,得到转码字符串。

[0079] 在本发明的一种具体实施方式中,Linux程序100,具体用于利用字符串格式化命令,将特殊字符串转为x+ASCII码形式,得到转码字符串。

[0080] 在本发明的一种具体实施方式中,目标脚本200,具体用于将转码字符串中的每一个x替换为\\x;利用字符串显示命令对转码字符串中的转码字符进行解码并更替,得到目标字符串。

[0081] 在本发明的一种具体实施方式中,Linux程序100,具体用于判断待传递字符串是否包括特殊字符;如果是,则将待传递字符串作为目标字符串。

[0082] 在本发明的一种具体实施方式中,目标脚本200,具体用于在调用目标脚本对转码字符串进行解码,得到目标字符串之后,利用目标字符串对目标文件进行修改。

[0083] 在本发明的一种具体实施方式中,目标脚本200,具体用于调用写文件工具,利用目标字符串对目标文件进行修改。

[0084] 实施例三:

[0085] 相应于上面的方法实施例,本发明实施例还提供了一种计算机,下文描述的一种计算机与上文描述的一种字符串传递方法可相互对应参照。

[0086] 参见图3所示,该计算机包括:

[0087] 存储器D1,用于存储计算机程序;

[0088] 处理器D2,用于执行计算机程序时实现上述方法实施例的字符串传递方法的步骤。

[0089] 具体的,请参考图4,为本实施例提供的一种计算机的具体结构示意图,该计算机可因配置或性能不同而产生比较大的差异,可以包括一个或一个以上处理器(central processing units,CPU)322(例如,一个或一个以上处理器)和存储器332,一个或一个以上存储应用程序342或数据344的存储介质330(例如一个或一个以上海量存储设备)。其中,存储器332和存储介质330可以是短暂存储或持久存储。存储在存储介质330的程序可以包括一个或一个以上模块(图示没标出),每个模块可以包括对数据处理设备中的一系列指令操作。更进一步地,中央处理器322可以设置为与存储介质330通信,在计算机301上执行存储介质330中的一系列指令操作。

[0090] 计算机301还可以包括一个或一个以上电源326,一个或一个以上有线或无线网络

接口350,一个或一个以上输入输出接口358,和/或,至少一个Linux操作系统341。

[0091] 上文所描述的字符串传递方法中的步骤可以由计算机的结构实现。

[0092] 实施例四:

[0093] 相应于上面的方法实施例,本发明实施例还提供了一种可读存储介质,下文描述的一种可读存储介质与上文描述的一种字符串传递方法可相互对应参照。

[0094] 一种可读存储介质,可读存储介质上存储有计算机程序,计算机程序被处理器执行时实现上述方法实施例的字符串传递方法的步骤。

[0095] 该可读存储介质具体可以为U盘、移动硬盘、只读存储器(Read-Only Memory, ROM)、随机存取存储器(Random Access Memory, RAM)、磁碟或者光盘等各种可存储程序代码的可读存储介质。

[0096] 专业人员还可以进一步意识到,结合本文中所公开的实施例描述的各示例的单元及算法步骤,能够以电子硬件、计算机软件或者二者的结合来实现,为了清楚地说明硬件和软件的可互换性,在上述说明中已经按照功能一般性地描述了各示例的组成及步骤。这些功能究竟以硬件还是软件方式来执行,取决于技术方案的特定应用和设计约束条件。专业技术人员可以对每个特定的应用来使用不同方法来实现所描述的功能,但是这种实现不应认为超出本发明的范围。

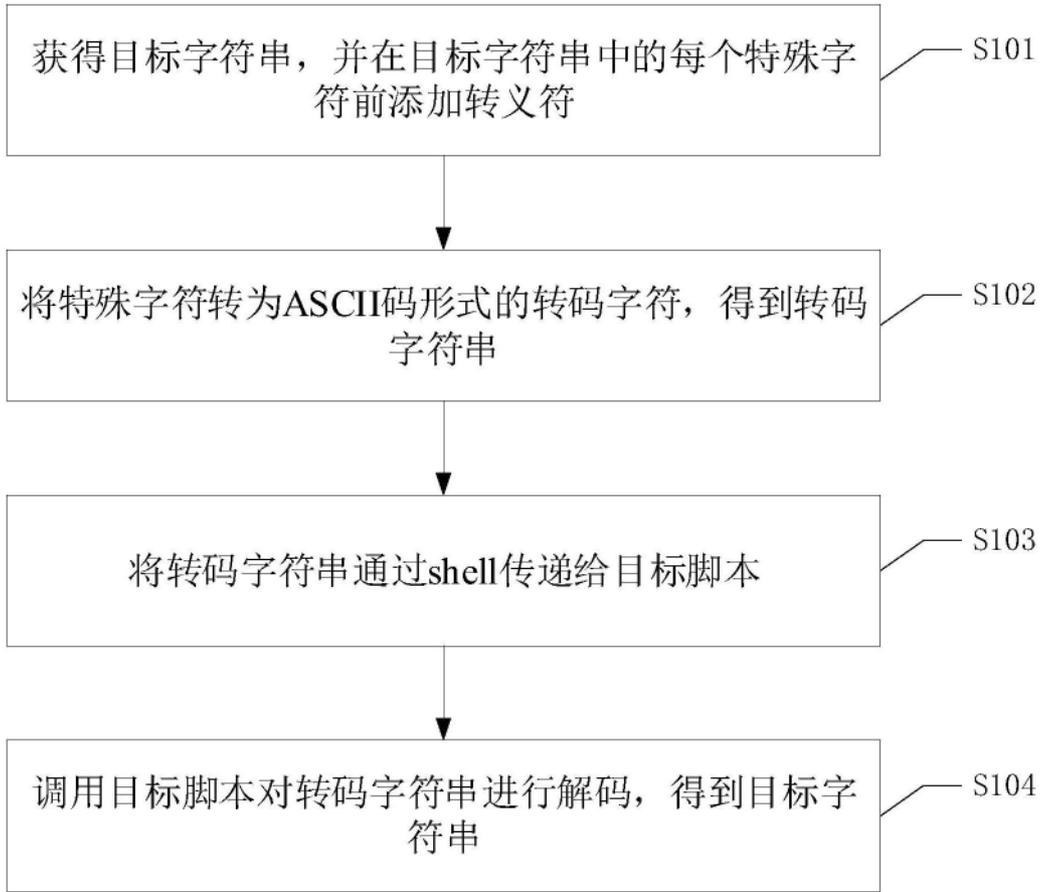


图1



图2

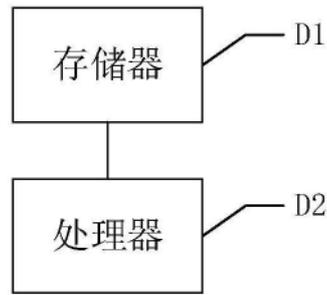


图3

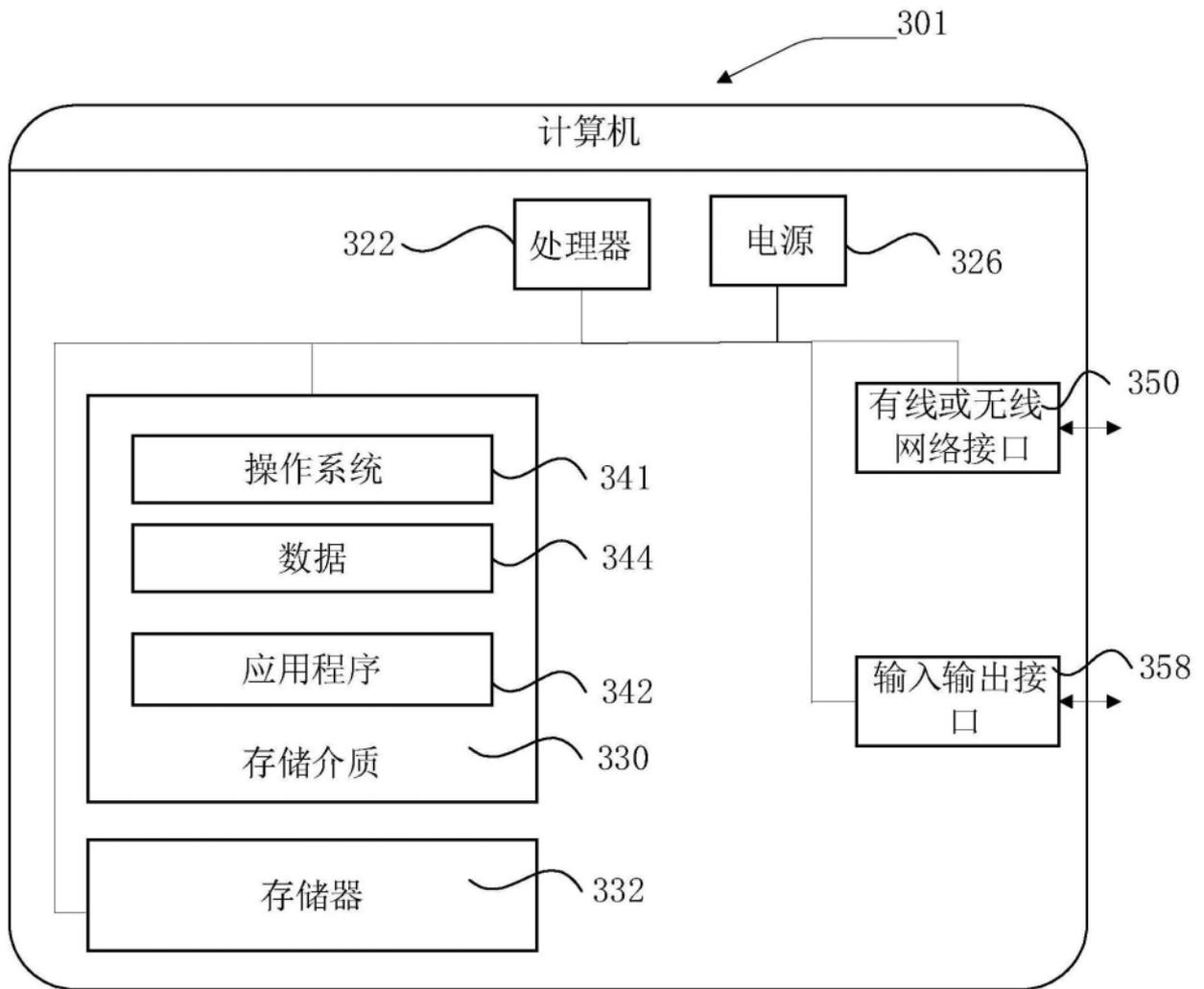


图4