(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 106843853 A
(43)申请公布日 2017.06.13

(21)申请号 201611239459.2

(22)申请日 2016.12.28

(71)申请人 北京五八信息技术有限公司
地址 100083 北京市海淀区学清路甲18号中关村东升科技园学院园三层301室

(72)发明人 吴朝彬 胡昊

(74)专利代理机构 北京同立钧成知识产权代理有限公司 11205
代理人 杨贝贝 刘芳

(51)Int.Cl.
G06F 9/44(2006.01)
G06F 21/12(2013.01)
G06F 21/14(2013.01)
H04L 29/06(2006.01)

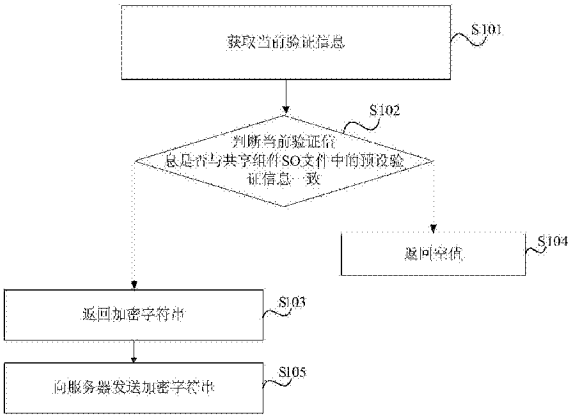
权利要求书1页 说明书5页 附图4页

(54)发明名称

保护用户信息的方法和装置

(57)摘要

本发明提供一种保护用户信息的方法和装置，通过将预设验证信息存储在SO文件中，在APP与服务器进行数据交互之前，APP端获取当前验证信息，确定验证信息是否与SO文件中的预设信息一致，如果当前验证信息与SO文件中的预设验证信息一致，则返回加密字符串。若当前验证信息与SO文件中的预设验证信息不一致，则返回空值；由于将预设验证信息封装在SO文件中，加大了破解难度，因此，破解者无法得到预设验证信息，进而，无法模拟网络请求，从而，保护用户信息，提高用户信息的安全性。



CN 106843853 A

1.一种保护用户信息的方法,其特征在于,包括:

获取当前验证信息;

判断所述当前验证信息是否与共享组件SO文件中的预设验证信息一致;

若所述当前验证信息与所述SO文件中的预设验证信息一致,则返回加密字符串;

若所述当前验证信息与所述SO文件中的预设验证信息不一致,则返回空值。

2.根据权利要求1所述的方法,其特征在于,还包括:

向服务器发送所述加密字符串。

3.根据权利要求2所述的方法,其特征在于,所述判断所述当前验证信息是否与SO文件中的预设验证信息一致之前,还包括:

获取所述预设验证信息;

将所述预设验证信息,封装到C语言或者C++语言实现的代码中,根据所述代码生成SO文件;

将所述SO文件存储在应用程序APP中。

4.根据权利要求3所述的方法,其特征在于,所述预设验证信息包括:APP的公钥和/或所述APP的签名文件。

5.根据权利要求4所述的方法,其特征在于,所述预设验证信息包括APP的公钥;

所述方法将所述预设验证信息,封装到C语言或者C++语言实现的代码中,根据所述代码生成SO文件之前,还包括:

与所述服务器协商所述APP公钥。

6.一种保护用户信息的装置,其特征在于,包括:

获取模块,用于获取当前验证信息;

处理模块,用于判断所述当前验证信息是否与共享组件SO文件中的预设验证信息一致;若所述当前验证信息与所述SO文件中的预设验证信息一致,则返回加密字符串;若所述当前验证信息与所述SO文件中的预设验证信息不一致,则返回空值。

7.根据权利要求6所述的装置,其特征在于,还包括:

发送模块,用于向服务器发送所述网络请求。

8.根据权利要求7所述的装置,其特征在于,所述获取模块还用于获取所述预设验证信息;

所述处理模块还用于将所述预设验证信息,封装到C语言或者C++语言实现的代码中,根据所述代码生成SO文件;将所述SO文件存储在应用程序APP中。

9.根据权利要求8所述的装置,其特征在于,所述预设验证信息包括:APP的公钥和/或所述APP的签名文件。

10.根据权利要求9所述的装置,其特征在于,所述预设验证信息包括APP的公钥;

所述获取模块还用于与所述服务器协商所述APP公钥。

保护用户信息的方法和装置

技术领域

[0001] 本发明涉及计算机技术，尤其涉及一种保护用户信息的方法和装置。

背景技术

[0002] 伴随着手机应用的飞速发展，电子设备的应用程序(Application，简称：APP)也得到了广泛应用。APP在电子设备中独立运行，通过与后台服务器进行数据交互，实现APP的功能。

[0003] 相关技术中，APP向服务器发送网络请求时，采用公钥对网络请求进行加密，服务器通过私钥对加密后的网络请求进行解密，校验网络请求中传输的参数的有效性。

[0004] 然而，采用现有技术的方法，如果应用程序被反编译，很容易被破解出公钥，破解者通过伪造公钥向后台服务器循环发起网络请求，暴力破解用户信息，因此，现有技术中用户信息的安全性不高。

发明内容

[0005] 本发明提供一种保护用户信息的方法和装置，以提高用户信息的安全性。

[0006] 本发明一个方面提供一种保护用户信息的方法，包括：

[0007] 获取当前验证信息；

[0008] 判断当前验证信息是否与共享组件SO文件中的预设验证信息一致；

[0009] 若当前验证信息与SO文件中的预设验证信息一致，则返回加密字符串；

[0010] 若当前验证信息与SO文件中的预设验证信息不一致，则返回空值。

[0011] 可选地，该方法还包括：

[0012] 向服务器发送网络请求。

[0013] 可选地，判断当前验证信息是否与SO文件中的预设验证信息一致之前，还包括：

[0014] 获取预设验证信息；

[0015] 将预设验证信息，封装到C语言或者C++语言实现的代码中，根据代码生成SO文件；

[0016] 将SO文件存储在应用程序APP中。

[0017] 可选地，预设验证信息包括：APP的公钥和/或APP的签名文件。

[0018] 可选地，预设验证信息包括APP的公钥；

[0019] 方法将预设验证信息，封装到C语言或者C++语言实现的代码中，根据代码生成SO文件之前，还包括：

[0020] 与服务器协商APP公钥。

[0021] 本发明另一个方面提供一种保护用户信息的装置，包括：

[0022] 获取模块，用于获取当前验证信息；

[0023] 处理模块，用于判断当前验证信息是否与共享组件SO文件中的预设验证信息一致；若当前验证信息与SO文件中的预设验证信息一致，则返回加密字符串；若当前验证信息与SO文件中的预设验证信息不一致，则返回空值。

[0024] 可选地，该装置还包括：

[0025] 发送模块，用于向服务器发送网络请求。

[0026] 可选地，获取模块还用于获取预设验证信息；

[0027] 处理模块还用于将预设验证信息，封装到C语言或者C++语言实现的代码中，根据代码生成SO文件；将SO文件存储在应用程序APP中。

[0028] 可选地，预设验证信息包括：APP的公钥和/或APP的签名文件。

[0029] 可选地，预设验证信息包括APP的公钥；

[0030] 获取模块还用于与服务器协商APP公钥。

[0031] 本发明提供的保护用户信息的方法和装置，通过将预设验证信息存储在SO文件中，在APP与服务器进行数据交互之前，APP端获取当前验证信息，确定验证信息是否与SO文件中的预设信息一致，如果当前验证信息与SO文件中的预设验证信息一致，则返回加密字符串。若当前验证信息与SO文件中的预设验证信息不一致，则返回空值；由于将预设验证信息封装在SO文件中，加大了破解难度，因此，破解者无法得到预设验证信息，进而，无法模拟网络请求，从而，保护用户信息，提高用户信息的安全性。

附图说明

[0032] 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性的前提下，还可以根据这些附图获得其他的附图。

[0033] 图1为本发明保护用户信息的方法实施例一的流程示意图；

[0034] 图2为本发明保护用户信息的方法实施例二的流程示意图；

[0035] 图3为本发明保护用户信息的方法实施例三的流程示意图；

[0036] 图4为本发明保护用户信息的装置实施例一的结构示意图。

具体实施方式

[0037] 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。

[0038] 本发明的说明书和权利要求书及上述附图中的术语“第一”、“第二”、“第三”“第四”等(如果存在)是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的数据在适当情况下可以互换，以便这里描述的本发明的实施例例如能够以除了在这里图示或描述的那些以外的顺序实施。此外，术语“包括”和“具有”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元的过程、方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

[0039] 本发明通过将预设验证信息存储在共享组件(Share Obiect，简称：SO)文件中，在APP与服务器进行数据交互之前，APP端先对当前验证信息进行验证，具体地，获取当前验证

信息，确定验证信息是否与SO文件中的预设信息一致，如果当前验证信息与SO文件中的预设验证信息一致，则返回加密字符串。若当前验证信息与SO文件中的预设验证信息不一致，则返回空值；由于将预设验证信息封装在SO文件中，加大了破解难度，因此，破解者无法得到预设验证信息，进而，无法模拟网络请求，从而，保护用户信息，提高用户信息的安全性。

[0040] 下面以具体地实施例对本发明的技术方案进行详细说明。下面这几个具体的实施例可以相互结合，对于相同或相似的概念或过程可能在某些实施例不再赘述。

[0041] 图1为本发明保护用户信息的方法实施例一的流程示意图，本实施例由APP端执行，本实施例的方法如下：

[0042] S101：获取当前验证信息。

[0043] 其中，当前验证信息通常与APP的账号唯一对应，例如：可以为APP的公钥、APP的签名文件或者其他信息。

[0044] S102：判断当前验证信息是否与共享组件SO文件中的预设验证信息一致；若是，则执行S103，若否，则执行S104。

[0045] 其中，预设验证信息包括但不限于：APP的公钥和/或APP的签名文件。

[0046] 其中，APP的公钥用于对向服务器发送的网络请求进行加密处理。

[0047] APP的签名文件，该文件在打包时编译生成，它唯一且不能重复，能够确保APP的唯一性。

[0048] 当判断当前验证信息与SO文件中的预设验证信息一致时，则执行S103。当判断当前验证信息与SO文件中的预设验证信息不一致时，说明当前验证信息的发送方应该是非法用户，则执行S104。

[0049] S103：返回加密字符串。

[0050] S104：返回空值。

[0051] S105：向服务器发送加密字符串。

[0052] 服务器通过私钥对加密字符串进行解密，然而进行验证，返回响应给APP端。

[0053] 举例来说：

[0054] 其中，APP的公钥一个示例如下所示：

[0055]

“MIICdwIBADANBgkqhksdfsfsdfsiG9w0BAQEFAASCAmEwggJdAgEAAoGBAMWRKAbbwylnEesVEi+XxIU7D0zjaxeUjkZdavJCIv6xFLumM6MkvAxeo4nXtrCdM/InlogbfUBEKiWl7HxmcRbWrjl0uso7A4SxwZhgynCxdfsH4cXQqt61GaLmbzIEvD8H1BazNCwerwerewrsdfhPDHdswkxKo0sdfsdfjrEuRh2h7+mfdsf9fcQPAgMBAAECgYEAv9F1eWM5kyM9ZG9q3saNyYVUD6TmP1i+41alFV1dOMeZh6+Qhx+NbL0eLyE/gS7AVqaTRkbaTuFfwnnsdfsdWi4757TY7QMdLHPN0LGeWDSA+/+UjkT8m4QyTAHV3hUjQd5195swo1GuuzLhCFhoYV+gW0kvWP3KsdfasfcbyiI7yedBpmECQQDq4n0DvD0isf76rotwxDo7OsRrbEsdfasfwerfXgxlQLsfsfsfsfsdfshMB/CG7bcmq3hZDbQ4sxzCqu2FQDBE8Mk5AsfkEA11PbucHT+0/Jy2sfSSG7BfwQ6Iy1hwRGiV/WLJIrL4ycdwdGFeNdzCV2ekCj1cb/cxTifL1Cdm9elWiSVBPfhwJBAMEOS0aIr49JgiYoXbIymPkeXhbmhTC9cvao8zYhAAPgb/sdfsfsfs/DGxw5pJQwY0c7A6S2IeTb5Oo0PLFAsIZr0Epuje9F0EqkCQEvq16L0Q/

AvY4m8PPikscYrCOsD05sL6jnPQJKNbXO+CwKgZe52VhG0Ze1Ru2dEK1JJLjym9GVLBm2CpMtaxFg=”

[0056] 消息摘要算法第五版(Message Digest Algorithm,简称:MD5)证书为例,如下所示:

[0057] “DC6DBD6E49682A57A8B82889043B93A8”

[0058] 本实施例,通过将预设验证信息存储在SO文件中,在APP与服务器进行数据交互之前,APP端获取当前验证信息,确定验证信息是否与SO文件中的预设信息一致,如果当前验证信息与SO文件中的预设验证信息一致,则返回加密字符串。若当前验证信息与SO文件中的预设验证信息不一致,则返回空值;由于将预设验证信息封装在SO文件中,加大了破解难度,因此,破解者无法得到预设验证信息,进而,无法模拟网络请求,从而,保护用户信息,提高用户信息的安全性。

[0059] 图2为本发明保护用户信息的方法实施例二的流程示意图,图2是在图1所示实施例的基础上,进一步地,在S102之前,还包括:

[0060] S1001:获取预设验证信息。

[0061] 预设验证信息包括但不限于下述至少一种:

[0062] APP公钥和APP的签名文件。

[0063] S1002:将预设验证信息,封装到C语言或者C++语言实现的代码中,根据代码生成SO文件。

[0064] 其中,一种可能的实现方式为:

[0065] 编写C语音或者C++语言实现的代码,实现判断当前验证信息是否与代码中的预设验证信息一致的逻辑,若一致,则根据预设验证信息中部分或者全部信息对网络请求进行加密,生成加密字符串。利用Java本地接口(Java Native Interface,简称:JNI)特征,使用本地语言开发包(native development kit,简称:NDK),通过Android.mk文件,将C语言或者C++语言代码编写成JAVA能识别的语言,最终生成SO文件。

[0066] 其中,SO文件可以根据电子设备的操作系统中的CPU型号的不同,可以分为很多种,例如:“armeabi-v7a”等,因其向下兼容的特性,只需要放在armeabi文件夹下即可。

[0067] S1003:将SO文件存储在应用程序APP中。

[0068] 其中,一种可能的实现方式为将SO文件存储在指定的目录,例如android工程中“main/jinLibs/armeabi”文件夹下,该目录通常是android工程指定的目录。该SO文件可以被JAVA代码加载。

[0069] 本实施例,通过获取预设验证信息,将预设验证信息,封装到C语言或者C++语言实现的代码中,根据代码生成SO文件,将SO文件存储在应用程序APP中。从而,实现将预设验证信息以二进制文件的形式存储在APP中,加大了破解难度,因此,破解者无法得到预设验证信息,进而,无法模拟网络请求,从而,保护用户信息,提高用户信息的安全性。

[0070] 图3为本发明保护用户信息的方法实施例三的流程示意图,图3是在图2所示实施例的基础上,当预设验证信息中包含APP的公钥时,在S1001之前,还包括:

[0071] S1000:APP与服务器协商APP公钥。

[0072] APP与服务器预先协商APP公钥,用于APP端对向服务器发送的请求加密;该公钥APP和服务器各自保存一份,服务器此外再多存储一份私钥,用于对APP发送的加密字符串

进行解密。

[0073] 本实施例，通过APP与服务器协商的方式，确定APP公钥，以便于APP与服务器进行数据交互的过程中进行加密处理，提高用户信息的安全性。

[0074] 图4为本发明保护用户信息的装置实施例一的结构示意图，本实施例的装置包括：获取模块401和处理模块402，其中，

[0075] 获取模块401用于获取当前验证信息；

[0076] 处理模块402用于判断当前验证信息是否与共享组件SO文件中的预设验证信息一致；若当前验证信息与SO文件中的预设验证信息一致，则返回加密字符串；若当前验证信息与SO文件中的预设验证信息不一致，则返回空值。

[0077] 其中，预设验证信息包括：APP的公钥和/或APP的签名文件。

[0078] 在图4中，还可以包括：发送模块403用于向服务器发送网络请求。

[0079] 本实施例，对应的可用于执行图1所示方法实施例的技术方案，其实现原理和技术效果类似，此处不再赘述。

[0080] 在保护用户信息的装置实施例一的基础上，本发明提供的保护用户信息的装置实施例二中，获取模块401还用于获取预设验证信息；处理模块402还用于将预设验证信息，封装到C语言或者C++语言实现的代码中，根据代码生成SO文件；将SO文件存储在应用程序APP中。

[0081] 本实施例，对应的可用于执行图2所示方法实施例的技术方案，其实现原理和技术效果类似，此处不再赘述。

[0082] 在保护用户信息的装置实施例一的基础上，本发明提供的保护用户信息的装置实施例二中，预设验证信息包括APP的公钥；获取模块401还用于与服务器协商APP公钥。

[0083] 本实施例，对应的可用于执行图3所示方法实施例的技术方案，其实现原理和技术效果类似，此处不再赘述。

[0084] 本领域普通技术人员可以理解：实现上述各方法实施例的全部或部分步骤可以通过程序指令相关的硬件来完成。前述的程序可以存储于一计算机可读取存储介质中。该程序在执行时，执行包括上述各方法实施例的步骤；而前述的存储介质包括：ROM、RAM、磁碟或者光盘等各种可以存储程序代码的介质。

[0085] 最后应说明的是：以上各实施例仅用以说明本发明的技术方案，而非对其限制；尽管参照前述各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，或者对其中部分或者全部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范围。

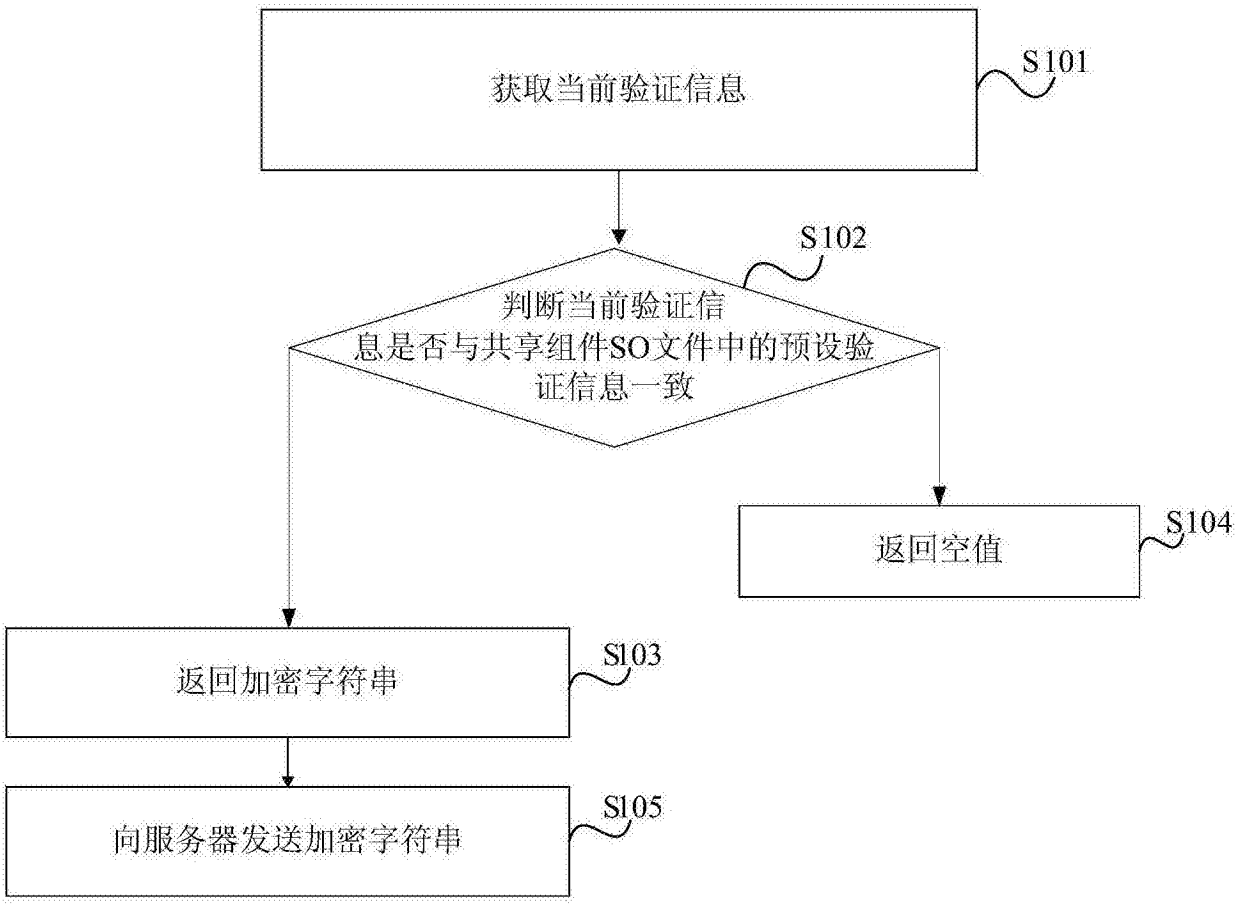

获取当前验证信息
S101
S102
判断当前验证信息是否与共享组件SO文件中的预设验证信息一致
返回空值
S104
返回加密字符串
S103
向服务器发送加密字符串
S105

图1

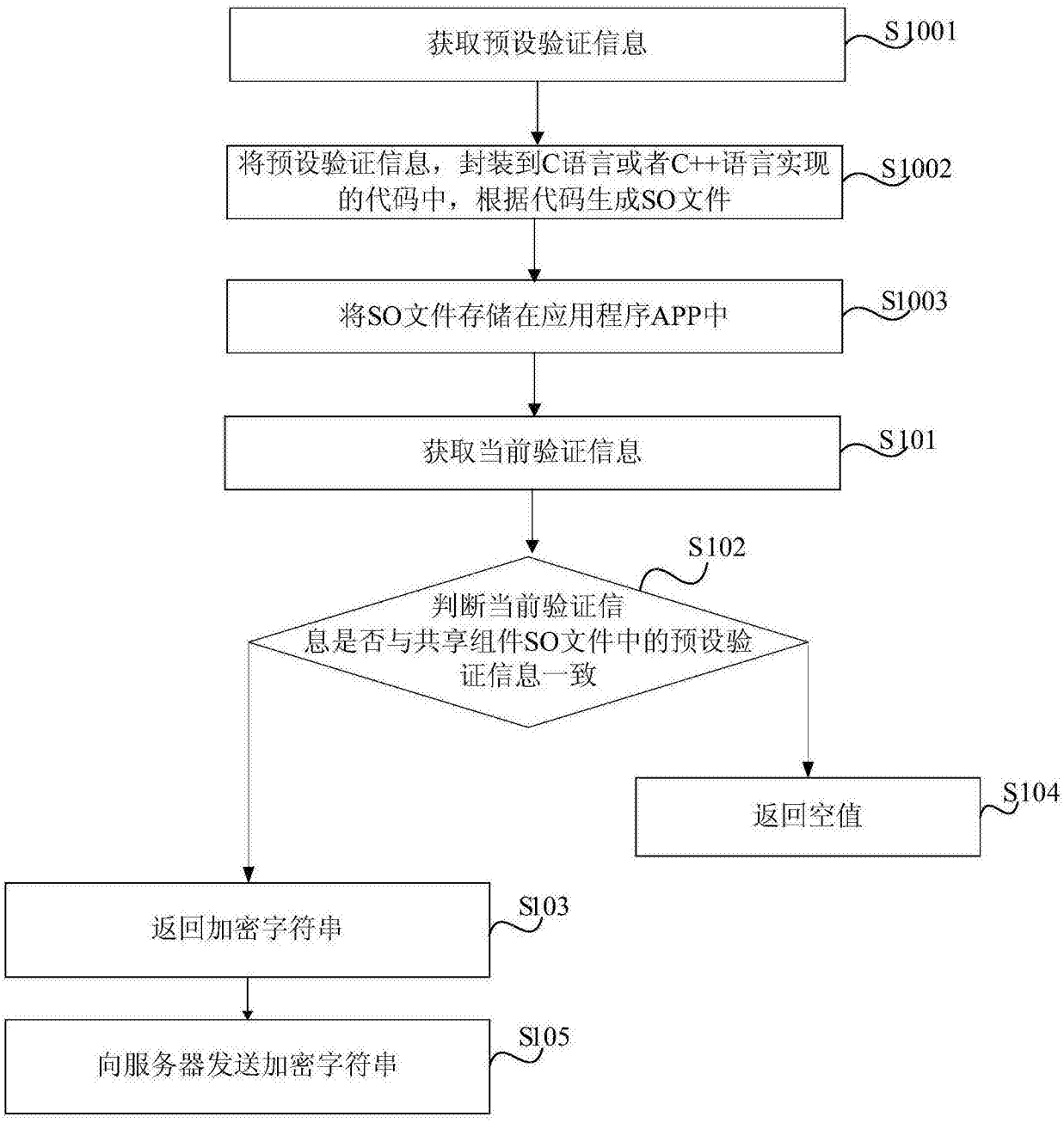

获取预设验证信息
S1001
将预设验证信息，封装到C语言或者C++语言实现的代码中，根据代码生成SO文件
S1002
将SO文件存储在应用程序APP中
S1003
获取当前验证信息
S101
S102
判断当前验证信息是否与共享组件SO文件中的预设验证信息一致
返回空值
S104
返回加密字符串
S103
向服务器发送加密字符串
S105

图2

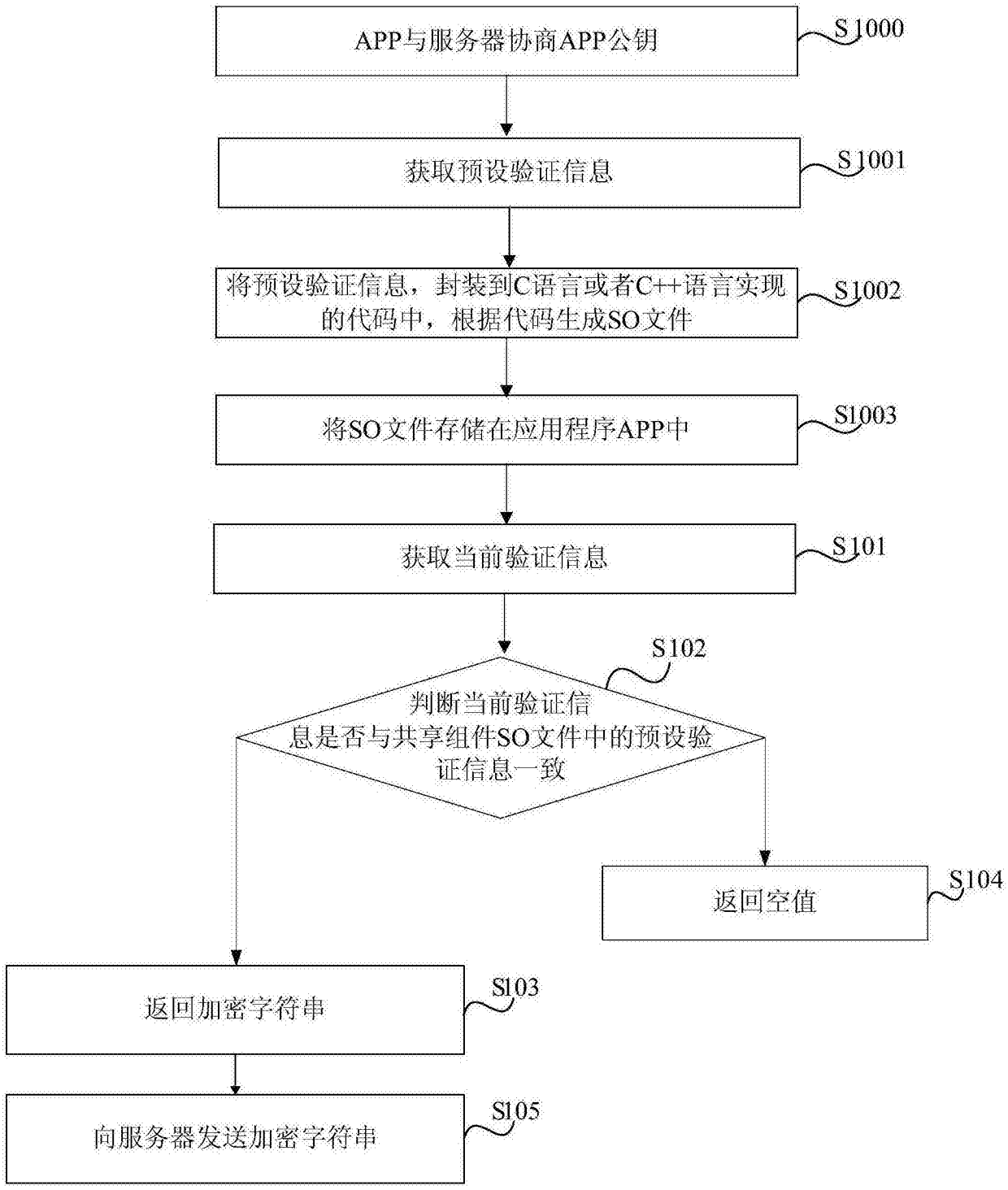

APP与服务器协商APP公钥
S1000
获取预设验证信息
S1001
将预设验证信息，封装到C语言或者C++语言实现的代码中，根据代码生成SO文件
S1002
将SO文件存储在应用程序APP中
S1003
获取当前验证信息
S101
判断当前验证信息是否与共享组件SO文件中的预设验证信息一致
S102
返回空值
S104
返回加密字符串
S103
向服务器发送加密字符串
S105

图3

图4