(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 101789065 A
(43)申请公布日 2010.07.28

(21)申请号 201010116359.7

(22)申请日 2006.02.13

(30)优先权数据
2005-036621 2005.02.14 JP

(62)分案原申请数据
200680000828.0 2006.02.13

(71)申请人 松下电器产业株式会社
地址 日本大阪府

(72)发明人 杰尔马诺·莱希森林 金丸智一

(74)专利代理机构 永新专利商标代理有限公司
72002
代理人 徐冰冰 黄剑锋

(51)Int.Cl.
G06F 21/00(2006.01)

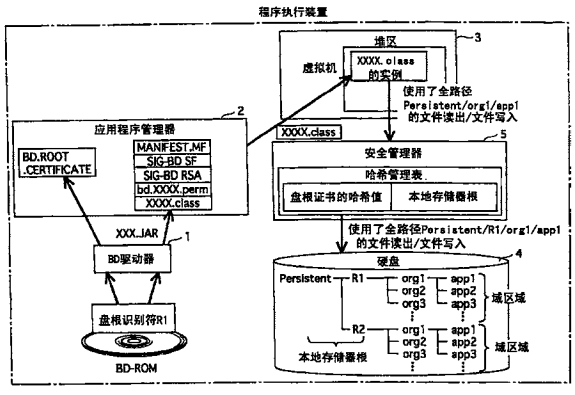
权利要求书 2 页 说明书 23 页 附图 16 页

(54)发明名称

应用程序执行装置、应用程序执行方法和程序

(57)摘要

本发明提供一种应用程序执行装置、应用程序执行方法和程序，BD-ROM上记录了盘根证书(301)。该盘根证书(301)将从根认证局发布的根证书分配给该盘介质。应用程序管理器(2)从盘根证书(301)中取得哈希值，并使用该哈希值来判断应用程序的正当性。在判断为正当的情况下，虚拟机(3)执行应用程序。另外，本地存储器(5)具有多个域区域，安全管理器(4)将本地存储器(5)内存在的多个域区域中、与所取得的哈希值对应的域区域分配给应用程序。



CN 101789065 A

1. 一种应用程序执行装置，从记录介质读出应用程序，并加以执行，所述记录介质记录有：盘根证书；第一根证书；应用程序；证书信息，包含提供者组织的提供者组织 ID 以及与作为对象的文件相关的信息，所述提供者组织将读出作为所述对象的文件的权限提供给所述应用程序；叶证书，包含与所述提供者组织 ID 相同的组织 ID；以及第二根证书，与所述叶证书相关联；所述应用程序执行装置的特征在于，

所述盘根证书是将从根认证局发布的第三根证书分配给该记录介质的证书，并且与作为所述对象的文件相关的信息是表示所述文件的地点的信息，还包含有与包含所述提供者组织 ID 的文件路径相关的信息；

该应用程序执行装置具有：

管理单元，通过判断从所述盘根证书取得的哈希值与从所述第一根证书取得的哈希值是否相同，来判断应用程序的正当性；

执行单元，在所述管理单元判断为正当的情况下，来执行所述应用程序；以及

存储单元，包含由文件路径确定的存储区域，所述文件路径使用了所述第二根证书的哈希值以及所述提供者组织 ID；

在所述执行中的应用程序指定包含所述提供者组织 ID 的文件路径，做出了对作为所述对象的文件进行读出的要求时，将该文件路径变换为使用了第二根证书的哈希值和所述指定的提供者组织 ID 的文件路径，并使所述应用程序读出作为由所述变换后的文件路径确定的存储区域内的对象的文件，所述第二根证书与包含和所述指定的提供者组织 ID 相同的组织 ID 的叶证书相关联。

2. 一种应用程序执行方法，使计算机执行应用程序，所述计算机具有存储单元，所述存储单元包含使用文件路径确定的存储区域，所述计算机构成为从记录介质读出应用程序，所述记录介质记录有：盘根证书；第一根证书；应用程序；证书信息，包含提供者组织的提供者组织 ID 以及与作为对象的文件相关的信息，所述提供者组织将读出作为所述对象的文件的权限提供给所述应用程序；叶证书，包含与所述提供者组织 ID 相同的组织 ID；以及第二根证书，与所述叶证书相关联；所述应用程序执行方法的特征在于，

所述盘根证书是将从根认证局发布的第三根证书分配给该记录介质的证书，并且与作为所述对象的文件相关的信息是表示所述文件的地点的信息，还包含有与包含所述提供者组织 ID 的文件路径相关的信息；

该应用程序执行方法使所述计算机执行如下步骤：

第一步骤，通过判断从所述盘根证书取得的哈希值与从所述第一根证书取得的哈希值是否相同，来判断应用程序的正当性；

第二步骤，在所述第一步骤中判断为正当的情况下，来执行所述应用程序；以及

第三步骤，在所述执行中的应用程序指定包含所述提供者组织 ID 的文件路径，做出了对作为所述对象的文件进行读出的要求时，将该文件路径变换为使用了第二根证书的哈希值和所述指定的提供者组织 ID 的文件路径，并使所述应用程序读出作为由所述变换后的文件路径确定的存储区域内的对象的文件，所述第二根证书与包含和所述指定的提供者组织 ID 相同的组织 ID 的叶证书相关联。

3. 一种程序，使计算机执行该程序，所述计算机具有存储单元，所述存储单元包含使用文件路径确定的存储区域，所述计算机构成为从记录介质读出应用程序，所述记录介质记

录有 :盘根证书 ;第一根证书 ;应用程序 ;证书信息,包含提供者组织的提供者组织 ID 以及与作为对象的文件相关的信息,所述提供者组织将读出作为所述对象的文件的权限提供给所述应用程序 ;叶证书,包含与所述提供者组织 ID 相同的组织 ID ;以及第二根证书,与所述叶证书相关联 ;所述程序的特征在于,

所述盘根证书是将从根认证局发布的第三根证书分配给该记录介质的证书,并且与作为所述对象的文件相关的信息是表示所述文件的地点的信息,还包含有与包含所述提供者组织 ID 的文件路径相关的信息 ;

该程序使所述计算机执行如下步骤 :

第一步骤,通过判断从所述盘根证书取得的哈希值与从所述第一根证书取得的哈希值是否相同,来判断应用程序的正当性 ;

第二步骤,在所述第一步骤中判断为正当的情况下,来执行所述应用程序 ;以及

第三步骤,在所述执行中的应用程序指定包含所述提供者组织 ID 的文件路径,做出了对作为所述对象的文件进行读出的要求时,将该文件路径变换为使用了第二根证书的哈希值和所述指定的提供者组织 ID 的文件路径,并使所述应用程序读出作为由所述变换后的文件路径确定的存储区域内的对象的文件,所述第二根证书与包含和所述指定的提供者组织 ID 相同的组织 ID 的叶证书相关联。

应用程序执行装置、应用程序执行方法和程序

[0001] 本发明是本申请人于2006年2月13日提交的中国专利申请号为200680000828.0、发明名称为“应用程序执行装置、管理方法和程序”的分案申请。

技术领域

[0002] 本发明是属于对应用程序进行资源分配的技术的领域的发明。

背景技术

[0003] 所谓资源分配技术属于将平台具有的存储装置（本地存储器）内的区域作为资源分配给应用程序技术的发明。为了体系化管理世界上各种组织提供的各种应用程序，作为面向欧洲的数字广播接收装置的MHP(Multimedia Home Platform)规定本地存储器的目录结构，使其可以以如下这种统一格式的文件路径，来加以访问。

[0004] 文件路径：Root/ 组织 ID/appID

[0005] 这里所谓组织 ID 是指唯一表示作为应用程序的提供源的组织的识别符，所谓 appID 是指唯一表示应用程序的识别符。

[0006] 每次实现以上的区域分配时，作为 MHP 的设备使用根证书，来判断应用程序的正当性。下面，对在设备 (MHP) 中作出的应用程序的正当性检查和对应用程序进行的区域分配来加以说明。

[0007] 创建应用程序的创建者每次将应用程序交付给设备时，在将根证书添加到应用程序后，将应用程序发送到设备。该根证书与固有分配给设备的根证书相同，设备接收添加了根证书的应用程序，判断在该应用程序上添加的根证书和分配给设备的根证书的一致性。若有一致性，则将与该应用程序的提供源的组织对应的组织用目录分配给该应用程序，而使应用程序进行该目录内的文件的访问。

[0008] 适当控制应用程序的访问权限的技术记载在非专利文献 1 中。

[0009] 非专利文献 1：“JAVA（注册商标）Security”Scott Oaks 著、O’Reilly 发行、May 2001、ISBN 0-596-00157-6

[0010] 但是，上述现有技术中，设置与应用程序的提供源的组织对应的区域，并将与该组织对应的目录属下的区域分配给要求访问的应用程序。因此，在应用程序执行装置遍布世界的所有区域，且从世界的所有组织提供应用程序的情况下，需要使世界各个组织的组织 ID 不重复的准备。这是因为，若不这样，属于某个组织的应用程序可以自由访问其他组织用的目录，不能保证应用程序使用的数据的机密性。由于需要该准备，所以需要对世界各组织分配唯一的组织 ID 的第三人机关，参加应用程序执行装置的标准化的制造商必须对该机关的设置·运营确保资金和人力。其对参加应用程序执行装置的标准化的制造商来说成为很大的负担。但是，若其他组织自由访问自身组织用的目录，则发生自身组织的应用程序用的数据被其他组织盗用、无授权使用的可能。如果装置中对该状况放任不管，则可以预计创建应用程序的创建者多发生不愿意向装置供给应用程序的情况。因此，有不希望发生的装置上操作的应用程序数量不充足，因应用程序不充足，有应用程序执行装置很难普及的问

题。

[0011] MHP 中，将分配给装置的根证书用于应用程序的正当性判断。在分配给装置的根证书被带有恶意的人暴露了的情况下，分配给设备的数字证书由装置的制造源更新为新的证明。在存在该数字证书的更新的情况下，将添加了旧的根证书的应用程序判断为正当性不正当，而不允许访问本地存储器。当然，由于如 MHP 中使用的应用程序那样，若应用程序是仅在广播时可使用的一次性应用程序，由于不断发送新的应用程序，所以这样处理可能就很充分了。但是，在应用程序进行 DVD-Video 内容、BD-ROM 内容等与磁盘上记录的视频作品有关的处理的情况下，要求无论哪次再现旧的视频作品，进行与该视频作品有关的处理的应用程序都要进行正确的操作。因此，绝不希望因数字证书的更新，应用程序不动作了。即，现有技术有在暴露了根证书的情况下，操作保障不彻底的问题。

发明内容

[0012] 本发明的第一目的是提供一种在世界范围内，即便不需要进行不发生重复的组织 ID 的管理，也可提高由各种组织供给的应用程序使用的文件的机密性的应用程序执行装置。

[0013] 本发明的第二目的是提供一种即使暴露了根证书的情况下，也要实现高水平的动作保障的应用程序执行装置。

[0014] 为了实现上述第一、第二目的，本发明的应用程序执行装置，从记录介质读出应用程序，并加以执行，所述记录介质记录有：盘根证书；第一根证书；应用程序；证书信息，包含提供者组织的提供者组织 ID 以及与作为对象的文件相关的信息，所述提供者组织将读出作为所述对象的文件的权限提供给所述应用程序；叶证书，包含与所述提供者组织 ID 相同的组织 ID；以及第二根证书，与所述叶证书相关联；所述应用程序执行装置的特征在于，所述盘根证书是将从根认证局发布的第三根证书分配给该记录介质的证书，并且与作为所述对象的文件相关的信息是表示所述文件的地点的信息，还包含有与包含所述提供者组织 ID 的文件路径相关的信息；该应用程序执行装置具有：管理单元，通过判断从所述盘根证书取得的哈希值与从所述第一根证书取得的哈希值是否相同，来判断应用程序的正当性；执行单元，在所述管理单元判断为正当的情况下，来执行所述应用程序；以及存储单元，包含由文件路径确定的存储区域，所述文件路径使用了所述第二根证书的哈希值以及所述提供者组织 ID；在所述执行中的应用程序指定包含所述提供者组织 ID 的文件路径，做出了对作为所述对象的文件进行读出的要求时，将该文件路径变换为使用了第二根证书的哈希值和所述指定的提供者组织 ID 的文件路径，并使所述应用程序读出作为由所述变换后的文件路径确定的存储区域内的对象的文件，所述第二根证书与包含和所述指定的提供者组织 ID 相同的组织 ID 的叶证书相关联。

[0015] 本发明的应用程序执行方法，使计算机执行应用程序，所述计算机具有存储单元，所述存储单元包含使用文件路径确定的存储区域，所述计算机构成为从记录介质读出应用程序，所述记录介质记录有：盘根证书；第一根证书；应用程序；证书信息，包含提供者组织的提供者组织 ID 以及与作为对象的文件相关的信息，所述提供者组织将读出作为所述对象的文件的权限提供给所述应用程序；叶证书，包含与所述提供者组织 ID 相同的组织 ID；以及第二根证书，与所述叶证书相关联；所述应用程序执行方法的特征在于，所述盘根证书

是将从根认证局发布的第三根证书分配给该记录介质的证书，并且与作为所述对象的文件相关的信息是表示所述文件的地点的信息，还包含有与包含所述提供者组织 ID 的文件路径相关的信息；该应用程序执行方法使所述计算机执行如下步骤：第一步骤，通过判断从所述盘根证书取得的哈希值与从所述第一根证书取得的哈希值是否相同，来判断应用程序的正当性；第二步骤，在所述第一步骤中判断为正当的情况下，来执行所述应用程序；以及第三步骤，在所述执行中的应用程序指定包含所述提供者组织 ID 的文件路径，做出了对作为所述对象的文件进行读出的要求时，将该文件路径变换为使用了第二根证书的哈希值和所述指定的提供者组织 ID 的文件路径，并使所述应用程序读出作为由所述变换后的文件路径确定的存储区域内的对象的文件，所述第二根证书与包含和所述指定的提供者组织 ID 相同的组织 ID 的叶证书相关联。

[0016] 本发明的程序，使计算机执行该程序，所述计算机具有存储单元，所述存储单元包含使用文件路径确定的存储区域，所述计算机构成为从记录介质读出应用程序，所述记录介质记录有：盘根证书；第一根证书；应用程序；证书信息，包含提供者组织的提供者组织 ID 以及与作为对象的文件相关的信息，所述提供者组织将读出作为所述对象的文件的权限提供给所述应用程序；叶证书，包含与所述提供者组织 ID 相同的组织 ID；以及第二根证书，与所述叶证书相关联；所述程序的特征在于，所述盘根证书是将从根认证局发布的第三根证书分配给该记录介质的证书，并且与作为所述对象的文件相关的信息是表示所述文件的地点的信息，还包含有与包含所述提供者组织 ID 的文件路径相关的信息；该程序使所述计算机执行如下步骤：第一步骤，通过判断从所述盘根证书取得的哈希值与从所述第一根证书取得的哈希值是否相同，来判断应用程序的正当性；第二步骤，在所述第一步骤中判断为正当的情况下，来执行所述应用程序；以及第三步骤，在所述执行中的应用程序指定包含所述提供者组织 ID 的文件路径，做出了对作为所述对象的文件进行读出的要求时，将该文件路径变换为使用了第二根证书的哈希值和所述指定的提供者组织 ID 的文件路径，并使所述应用程序读出作为由所述变换后的文件路径确定的存储区域内的对象的文件，所述第二根证书与包含和所述指定的提供者组织 ID 相同的组织 ID 的叶证书相关联。

[0017] 另外，本发明的应用程序 (application) 执行装置，从记录有盘根证书和应用程序的盘介质读出应用程序，并加以执行，其特征在于：盘根证书是该盘介质的创建者将从根认证局发布的根证书分配给该盘介质的证书；该应用程序执行装置具有：管理单元，从盘根证书取得哈希值，并使用该哈希值来判断应用程序的正当性；执行单元，在管理单元判断为正当的情况下，来执行应用程序；存储单元，具有多个域区域；以及分配单元，将存储单元内存在的多个域区域中与所取得的哈希值对应的域区域分配给应用程序。

[0018] 发明效果

[0019] 本发明的应用程序执行装置由于具有上述这种技术事项 (1)，所以在本地存储器内部存在多个域区域，各个域区域分别分配根证书哈希值。并且，若在这些域区域下，按每个组织，生成每个组织的区域，则在世界范围中，组织 ID 可以不唯一。在“域区域”这种封闭的世界中，可区别多个组织就足够了，所以组织 ID 在世界范围中不需要是唯一的值，不需要基于第三人机关的管理。即便不进行使组织 ID 不重合的管理，也可在一个平台中使从世界中的组织供给的应用程序动作，同时，可以提高作为应用程序进行读出 / 写入的对象的文件的机密性。

[0020] 即便不需要基于第三人机关的管理，创建者也会消除不愿意提供应用程序的障碍，可以提高应用程序使用的文件的独立性、机密性，所以可以呼吁视频创作者、视频分配者、广播台、出版者、软件工作室等世界中的多个组织进行应用程序执行装置用应用程序的提供。由此，可以实现应用程序的丰富化，可以充实装置用应用程序，可以进一步提高作为盘介质再现装置的应用程序执行装置的普及。

[0021] 根证书提供给存储单元内的域区域，而不是装置主体。若由某个盘介质供给的应用程序与该盘介质的盘根证书对应，则只要将该盘介质装到应用程序执行装置中，则必然保障了动作。尤其，并非没有可能暴露盘根证书，但是这种情况下，不可使用该盘介质，或仅更新对该盘介质的盘根证书就可以了，由其他盘供给的应用程序如现有技术那样，使用盘根证书就可以了，所以可以实现可靠的动作保障。

[0022] 由于不需要这种世界范围中的组织 ID 的管理，而可以将现有技术的应用程序的彼此保障维持在高水平，所以本发明的应用程序执行装置对执行进行与视频作品有关的处理的应用程序的应用程序执行装置的世界标准作出了大的贡献。

[0023] 虽然是任意，但是上述的应用程序执行装置的技术事项 (1) 通过添加下面的技术事项 (2)(3)，具体构成应用程序执行装置，从而可以带来进一步的效果。本申请中，将这些技术事项区分为权利要求 1 下的从属形式的权利要求来加以记载。

[0024] • 技术事项 (2) 在盘介质上记录有第一数字证书链；所述管理单元进行的正当性判断包含：从所述第一数字证书链中的根证书取得的哈希值和从所述盘根证书取得的哈希值是否一致的检查；所述域区域包含多个组织区域；所述分配单元进行的分配包含：在哈希值一致的情况下，许可应用程序使用被分配给应用程序的域区域中的多个组织区域中、与第一数字证书链中的叶证书上记载的组织 ID 对应的组织区域。

[0025] 根据在应用程序执行装置上添加上述的技术事项，可以使用利用了现有的认证局的商业模块，同时，不能不正当使用第一应用程序向第二应用程序分配的资源。防止该不正当使用通过多个组织共用相同的盘根证书来进行，由于在应用程序执行装置内没有保持根证书，所以可以维持更新了应用程序执行装置中的情况下的动作互换。

[0026] • 技术事项 (3) 作为所述应用程序的供给源的组织从其他组织接受区域使用权限的提供；所述盘介质上进一步记录有证书信息，该证书信息包含提供者组织 ID 和接受者组织 ID，该提供者组织 ID 表示作为区域使用权限的提供者的组织，该接受者组织 ID 表示作为区域使用权限的接受者的组织；所述管理单元进一步进行所述证书信息的正当性的确认；由所述分配单元进行的分配包含：在确认了所述证书信息的正当性的情况下，允许应用程序使用被分配给应用程序的域区域中的多个组织区域中、与提供者组织 ID 对应的组织区域。

[0027] • 技术事项 (4) 所述证书信息包含从提供者组织所固有的根证书取得的哈希值和从接受者组织所固有的根证书取得的哈希值；所述管理单元进行的证书信息的正当性判断包含：从盘根证书取得的哈希值与从接受者组织所固有的根证书取得的哈希值是否一致的检查；以及所述第一数字证书链条中的叶证书上所记载的组织 ID 和证书信息上所表示的接受者组织 ID 是否一致的检查。

[0028] 在应用程序执行装置上添加了上述的技术事项后，对应用程序执行装置可以重复进行如下的盘介质的装载、排出，在这多个盘介质上分配了不同的盘根证书的情况下，与各

个应用程序对应的盘根证书不同的情况下，也可访问将在某个盘介质上记录的第一应用程序分配给在另一盘介质上记录的第二应用程序的域区域。这样，在应用程序记录在不同的盘介质上，所分配的盘根证书不同的情况下，应用程序可以共用多个域区域。由此，可以提高创建者相互之间的联合、协调性。

[0029] • 技术事项 (5) 在盘介质上记录有第二数字证书链；所述管理单元进行的证书信息的正当性判断进一步包含：从所述第二数字证书链中的根证书中取得哈希值，该哈希值与从所述提供者组织所固有的根证书中取得哈希值是否一致的检查；以及所述第二数字证书链中的叶证书上所记载的组织 ID 与提供者组织 ID 是否一致的检查。

[0030] 在将上述这种技术事项添加到应用程序执行装置中时，可以防止证书信息的窜改，并且，由于在没有证书信息时，不能向其他应用程序用的域区域进行访问，所以可以防止不正当的访问。

[0031] • 技术事项 (6) 所述证书信息进一步包含文件列表；所述分配单元进行的组织区域的使用许可包含：使应用程序访问在组织区域下存在的文件中由文件列表表示的文件。

[0032] 在将上述这种技术事项添加到应用程序执行装置中时，可以精细进行基于证书信息的访问提供，可以抑制由应用程序不合适造成的数据的破坏危险。

[0033] • 技术事项 (7) 所述证书信息进一步具有表示文件的访问方法的访问信息；所述分配单元进行的组织区域的使用许可包含：以访问信息所表示的访问方法，使所述应用程序访问组织区域下存在的文件中由文件列表表示的文件。

[0034] 在将上述这种技术事项添加到应用程序执行装置中时，在存在基于使用了证书信息的应用程序的访问时，也可抑制由应用程序不合适造成的数据的破坏危险。

[0035] • 技术事项 (8) 在盘介质上记录有第一数字证书链；所述管理单元进行的正当性判断包含：从所述第一数字证书链内的根证书取得的哈希值与从所述盘根证书取得的哈希值是否一致的检查；在哈希值一致的情况下进行所述执行装置进行的应用程序的执行。

[0036] 在将上述这种技术事项添加到应用程序执行装置中时，可以防止恶意进行应用程序之间的通信 (Interprocess Communication、IP) 的攻击，可以提高安全性。

附图说明

[0037] 图 1(a) 是表示 BD-ROM 中的文件 • 目录结构的图；(b) 是表示 Java(TM) 存档文件 302 中的结构的一例的图。

[0038] 图 2(a) 是表示 Credential(证书) 的数据结构的一例的图，(b) 是表示 Credential 的具体一例的的图。

[0039] 图 3(a) 是模式表示了在 BD-ROM 中怎样分配根证书的图；(b) 是模式表示在 MHP 中怎样分配根证书的图。

[0040] 图 4 是表示没有提供权限时的 SIG-BD.RSA、SIG-BD.SF、BD.ROOT.CERTIFICATE、MANIFEST.MF 的相互关系的图。

[0041] 图 5 是表示提供了权限时的 SIG-BD.RSA、SIG-BD.SF、BD.ROOT.CERTIFICATE、MANIFEST.MF、bd.XXXX.perm 的相互关系的图。

[0042] 图 6 是表示本实施形态中的应用程序执行装置的功能结构框图。

[0043] 图 7 是表示应用程序管理器 2 进行的根据 Java(TM) 存档文件 302 内的类文件的

应用程序的启动顺序的流程图。

[0044] 图 8 是表示应用程序管理器 2 进行的 Credential 的签名验证的顺序的流程图。

[0045] 图 9 是表示应用程序管理器 2 保持的管理信息的一例的图。

[0046] 图 10 是表示应用程序管理器 2 保持的管理信息的一例的图。

[0047] 图 11 是表示 Java(TM) 应用程序使用硬盘 4 时的处理顺序的流程图；

[0048] 图 12 是表示安全管理器 5 进行的本地存储器名的取得顺序的细节流程图。

[0049] 图 13 是表示安全管理器 5 保持的哈希管理表格的一例的图。

[0050] 图 14 是表示安全管理器 5 进行的组织 ID 的取得函数的细节的流程图。

[0051] 图 15 是表示安全管理器 5 进行的文件读出函数的细节的流程图。

[0052] 图 16 是表示安全管理器 5 进行的文件读出函数的细节的流程图。

[0053] 图 17 是表示安全管理器 5 进行的文件读入函数的细节的流程图。

[0054] 图 18 是表示安全管理器 5 进行的文件写入函数的细节的流程图。

[0055] 标记说明

[0056] 1BD 驱动器

[0057] 2 应用程序管理器

[0058] 3 虚拟机

[0059] 4 硬盘

[0060] 5 安全管理器

[0061] 301 盘根证书

[0062] 302Java(TM) 存档文件

[0063] 401 类文件

[0064] 402 清单文件

[0065] 403 签名文件

[0066] 404 数字签名文件

[0067] 405 允许请求文件

[0068] 501 提供者根证书的哈希值

[0069] 502 提供者组织 ID

[0070] 503 接受者根证书的哈希值

[0071] 504 接受者组织 ID

[0072] 505 接受者应用程序 ID

[0073] 506 文件列表

具体实施方式

[0074] 下面，参考附图来说明本发明的实施形态。

[0075] （第一实施形态）

[0076] 之后，说明本发明的应用程序执行装置的实施形态。首先，开始说明对应用程序执行装置供给应用程序的记录介质。作为该记录介质，本实施形态中，选 BD-ROM 作为题材。这是因为 BD-ROM 中的应用程序进行与如上所述的视频作品有关的处理。图 1(a) 是表示 BD-ROM 中的文件・目录结构的图。该图的第一级表示 BD-ROM。BD-ROM 与其他光盘、例如

DVD 或 CD 等相同，从其内周到外周按螺旋状具有记录区域。第二级表示该记录区域。如第二级所示，记录区域在内周的“导入区域”和外圈的“导出区域”之间具有可记录逻辑数据的“逻辑地址空间”。另外，在导入的内侧有仅由称作 BCA(Burst Cutting Area) 的驱动器来读出的特别区域。由于该区域不从应用程序中读出，所以例如可很好地用于著作权保护技术等。

[0077] “逻辑地址空间”上，以文件系统信息（卷）为开头，记录了视频数据、类文件及其关联信息所在的 Java(TM) 存档文件 302 等的数据。所谓文件系统是指 UDF(Universal Disk Format) 或 ISO9660 等，可以使用目录、文件结构来读出如与通常的个人计算机相同那样记录的逻辑数据。第三级表示 BD-ROM 的目录·文件结构。该目录·文件结构在根目录 (ROOT) 之下放置 BDDATA 目录。在目录 BDDATA 中记录了下面两种文件。

[0078] (A)BD.ROOT.CERTIFICATE :盘根证书 301

[0079] 所谓盘根证书 301 是指创建该 BD-ROM 的创建者将从根认证局接收了发布的根证书分配给 BD-ROM。盘根证书 301 例如以 X.509 的形式来进行编码。X.509 的标准记载在由国际电信电话咨询委员会发行的 CCITTRecommendation X.509(1988)，“The Directory-Authentication Framework”中。将该根证书记录在可移动型的记录介质上暴露根证书的可能性高，DVD 中，认为该分配有暴露根证书的危险，而不导入。但是，BD-ROM 中采用了与 DVD 相比，相当高级的著作权保护技术，该著作权保护技术的采用对导入“对 BD-ROM 分配固有的根证书”的考虑起到保障。应注意 BD-ROM 中盘根证书 301 的导入有如上这种背景。

[0080] (B)XXX.JAR :Java(TM) 存档文件 302

[0081] 这是基于 http://java(TM).sum.com/j2se/1.4.2/docs/guide/jar/jar.html 上记载的标准的 Java(TM) 存档文件 302。Java(TM) 存档文件 302 将 ZIP 文件的形式具体化为 Java(TM)，可以通过市场上销售的 ZIP 解压缩软件来确认内容。这里“XXX”可变，扩展符“JAR”固定。

[0082] Java(TM) 存档文件 302 以目录结构的形式来存储多个文件。图 1(b) 是表示 Java(TM) 存档文件 302 中的结构的一例的图。

[0083] 该结构在根目录下存在 XXXX.class，在 META-INF 目录下存在文件 MANIFEST.MF、文件 SIG-BD.SF、文件 SIG-BD.RSA、文件 bd.XXXX.perm。下面，分别说明这些文件。

[0084] (i)XXXX.class :类文件 401

[0085] 类文件 401 是存储了定义可在虚拟机上执行的 Java(TM) 应用程序这种结构体的类文件 401。

[0086] 由该类文件 401 定义的 Java(TM) 应用程序是通过 Xlet 接口，由应用程序执行装置的应用程序管理器，来加以控制的 Java(TM)Xlet。Xlet 接口具有“loaded(装载)”、“paused(暂停)”、“active(激活)”、“destoryed(破坏)”四种状态。

[0087] (ii)MAIFEST.MF :清单文件 402

[0088] 清单文件 402 对应于数字证书，是记载了 Java(TM) 存档文件 302 的属性、Java(TM) 存档文件 302 内的类文件 401 或数据文件的哈希值的文件。Java(TM) 存档文件 302 的属性有作为类文件 401 的实例的在 Java(TM) 应用程序上添加的应用程序 ID 和为执行 Java(TM) 存档文件 302 最初应执行的类文件 401 名。在不存在上述两个 Java(TM) 存

档文件 302 的属性的情况下，不执行 Java(TM) 存档文件 302 中作为类文件 401 的实例的 Java(TM) 应用程序。

[0089] (iii)SIG-BD.SF :签名文件 403

[0090] 签名文件 403 是记载了清单文件 402 的哈希值的文件。

[0091] (iv)SIG-BD.RSA :数字签名文件 404

[0092] 数字签名文件 404 是记载了“数字证书链”、签名文件 403 的“签名信息”的文件。

[0093] 签名文件 403 中的“签名信息”可以通过对签名文件 403 实施签名处理来得到。这些签名处理使用与数字签名文件 404 内的数字证书链中的公钥对应的密钥。

[0094] 所谓“数字证书链”是指具有如下形式的多个证书群 :第一证书（根证书）签名第二证书，并与此相同，第 n 证书签名第 n+1 证书。将数字证书链的最后的证书称作“叶证书”。通过使用该结构，通过从根证书起依次保证下一证书，而可以保障到数字证书链的最后证书。

[0095] “根证书”存储与 BD.ROOT.CERTIFICATE 文件中存在的盘根证书 301 相同的证书。

[0096] “叶证书”记载组织 ID。签名文件 403 以 PKCS#7 的形式来加以存储。PKCS#7 是存储签名和一个以上的数字证书用的文件形式，记载在由 IETF(Internet Engineering Task Force) 发行的 RFC2315 中。RFC2315 可以通过 http://www.ietf.org/rfc/rfc2315.txt 来加以参考。

[0097] 通常，该数字证书链是一个，但是提供后述的权限的情况下，生成两个该数字证书链。将这两个数字证书链称作第一数字证书链和第二数字证书链。第一数字证书链的根证书表示接收提供权限侧的组织的盘根证书，叶证书表示接收提供权限侧的组织的组织 ID。第二数字证书链的根证书表示提供权限侧的组织的盘根证书，叶证书表示提供权限侧的组织的组织 ID。另一方面，在不提供权限的情况下，数字证书链仅为一个（第一数字证书链）。

[0098] 清单文件 402、签名文件 403 和数字签名文件 404 的细节记载在 Java(TM) 存档文件的标准中。清单文件 402、签名文件 403 和数字签名文件 404 用于进行签名处理和签名验证处理。最终可以通过数字证书来签名 Java(TM) 存档文件 302 中的作为类文件 401 的实例的 Java(TM) 应用程序或允许请求文件 405。之后，将清单文件 402、签名文件 403 和数字签名文件 404 统称作“基于数字证书的签名”。

[0099] (v)bd.XXXX.perm :允许请求文件 405

[0100] 允许请求文件 405 是存储对所执行的 Java(TM) 应用程序提供哪种允许的信息的文件。具体存储下面的信息。

[0101] （ア）证书（证书）(数字信用证书）

[0102] （イ）应用程序之间通信的允许信息

[0103] 下面，说明（ア）证书。所谓“Credential(证书)”是指共享属于某个组织的组织目录内的文件用的信息。该共享通过将使用属于某个组织的应用程序用文件的权限提供给属于其他组织的应用程序来进行。因此，Credential 包含表示提供权限侧的组织的提供者组织 ID、表示接收权限侧的组织的识别的接收侧组织 ID。

[0104] 图 2(a) 表示 Credential 的数据结构的一例。在 Credential 中包括从根认证局向提供者组织发布的根证书的哈希值 501、对提供者组织分配的提供者组织 ID502、从根认证局向接受者发布的接受者根证书的哈希值 503、对接受者组织分配的接受者组织 ID504、

接受者应用程序 ID505 和提供文件列表 506。提供文件列表 506 中存储了一个以上的提供文件名 507 和访问方法 508（可读取 •可写入）的信息。为了使 Credential 有效必须进行签名。Credential 的签名与数字签名文件 404 相同，可以使用 PKCS#7 的方式。

[0105] 图 2(b) 是表示 Credential 的具体的一例。该图中的 Credential 中，通过 Credential 对文件“4/5/scores.txt”提供允许读取，对文件“4/5/etc/settings.txt”提供允许读写。

[0106] 接着，说明（イ）应用程序之间通信的允许信息。一个 Java(TM) 存档文件 302 中包含的 Java(TM) 应用程序通常不允许与其他 Java(TM) 存档文件 302 中包含的 Java(TM) 应用程序通信（应用程序之间通信）。仅在允许请求文件 405 中提供了允许应用程序之间通信的情况下可进行应用程序之间的通信。

[0107] 以上，是对于允许请求文件 405 的说明。接着，进行根证书的更详细的说明。

[0108] 图 3(a) 是模式表示了在 BD-ROM 中怎样分配根证书的图。该图的第三级表示设备（应用程序执行装置）和在该设备上装入的 BD-ROM，第二级表示创建这些设备和 BD-RM 的从业者 (BD-ROM 的创建者、设备的制造者）。第一级表示管理根证书的根认证局。

[0109] 该图中，BD-ROM 的创建者从根认证局中接收根证书的发布 (f1)，将所发布的根证书作为盘根证书 301 分配给 BD-ROM 后，将该根证书存储到 BD.ROOT.CERTIFICATE 中，而写入到 BD-ROM 中 (w1)。另一方面，每次生成 Java(TM) 存档文件 302 时，将该根证书和表示组织 ID 的叶证书收录到 SIG-BD.SF 中，而使其包含在 Java(TM) 存档文件 302 中。

[0110] 虽然不是本发明的本实施形态的说明，但是为了进行对比，说明 MHP 中的根证书的分配。

[0111] 图 3(b) 是表示在 MHP 中怎样分配根证书的图。MHP 中，制造设备的制造者从根认证局接收根证书的分发 (f2)，该制造者将该根证书分配给设备 (w2)。另一方面，广播内容的创作者将该根证书和表示自身的组织 ID 的叶证书添加到定义应用程序的类文件上后，送到设备中。若比较这些图，则可以看出 MHP 中将分配给设备的根证书分配给 BD-ROM，并从分配给该 BD-ROM 的根证书中形成证书链。

[0112] 不从 BD-ROM，而使 Java(TM) 存档文件 302 从 WWW 服务器中下载，并写入到应用程序执行装置内的存储装置中的情况也相同。该下载的目的是更新 BD-ROM 上记录的内容，但是在该下载中，将与容纳在 BD.ROOT.CERTIFICATE 中，与作为盘根证书 301 写入的根证书具有一致性的根证书存储到 SIG-BD.SF 中，而使其包含在 Java(TM) 存档文件中。由此，在将目的为更新 BD-ROM 中记录的内容的 Java(TM) 存档文件 302 通过下载供给应用程序执行装置的情况下，也可以通过使用对 BD-ROM 分配的盘根证书 301，而使应用程序执行装置确认 Java(TM) 存档文件 302 的正当性。

[0113] 图 4 是表示没有提供权限的情况下的 SIG-BD.SF、BD.ROOT.CERTIFICATE 的相互关系的图。该图中的箭头 d1 表示这些文件的内部结构的信息要素中有一致性。在没有提供权限的情况下，BD.ROOT.CERTIFICATE 内的根证书（盘根证书 301）与 SIG-BD.RSA 中的第一数字证书链内的根证书具有一致性。

[0114] 由于 MANIFEST.MF 签名类文件 XXXX.class，SIG-BD.SF 包含从 MANIFEST.MF 算出的哈希值，SIG-BD.RSA 包含从 SIG-BD.SF 算出的哈希值（箭头 h1），所以通过确认这些签名是否正当，判断这些图中所示的一致性，从而应用程序执行装置可以判断 Java(TM) 存档

文件 302 是否正当，是否施加了窜改。由于假定为没有提供权限，所以在该图中，没有表示 bd.XXXX.perm。

[0115] 图 5 是表示提供权限的情况下的 SIG-BD.RSA、SIG-BD.SF、BD.ROOT.CERTIFICATE、bd.XXXX.perm 的相互关系的图。该图中的箭头 d1 ～ d6 表示这些文件的内部结构的信息要素中有一致性。在提供了权限的提供权限情况下，BD.ROOT.CERTIFICATE 内的根证书（盘根证书）与 SIG-BD.RSA 中的第一数字证书链内的根证书具有一致性（箭头 d1）。另一方面，若提供了权限，则由于 BD.ROOT.CERTIFICATE 内的盘根证书 301 是接受者的证书，所以 bd.XXXX.perm 中的 Credential 的接受者根证书与 BD.ROOT.CERTIFICATE 内的根证书具有一致性（箭头 d2）。另外，Credential 中的接受者组织 ID 与第一数字证书链中的叶的组织 ID 具有一致性（箭头 d3）。

[0116] bd.XXXX.perm 中的 Credential 的提供者组织的根证书与 SIG-BD.RSA 内的第二数字证书链中的根证书具有一致性（箭头 d4）。另外，证书中的提供者组织 ID 与 SIG-BD.RSA 的第二数字证书链中的叶的组织 ID 具有一致性（箭头 d5）。Credential 中的接受者 ID 与 bd.XXXX.perm 中 Credential 之外的部分存在的应用程序 ID 具有一致性（箭头 d6）。

[0117] 由于 MANIFEST.MF 包含从类文件 XXXX.class 算出的哈希值，SIG-BD.SF 包含从 MANIFEST.M 算出的哈希值，SIG-BD.RSA 包含从 SIG-BD.SF 取得的哈希值（箭头 h1），所以通过确认这些签名是否正确，并判断这些图中所示的一致性，应用程序执行装置可以判断 Java(TM) 存档文件 302 是否正当，是否施加了窜改。虽然可预先决定，但是在本实施形态中，比较对各个根证书算出的哈希值，并根据这些哈希值是否一致来判断根证书的一致性。可以一次进行哈希值的运算，一般进行将所算出的哈希值存储到存储器等中，来加以使用。将从根证书中算出哈希值和取出存储器中存储的哈希值称作哈希值的“取得”。

[0118] 以上是对 BD-ROM 的说明。接着，说明本发明中的应用程序执行装置的内部结构。

[0119] 本实施形态的应用程序执行装置通过在具有 CPU、ROM、RAM、硬盘驱动器、BD-ROM 驱动器、AV 解码器、输入输出设备等的计算机系统上完全安装 Java(TM)2Micro_Edition(J2ME)Personal Basis Profile(PBP 1.0) 和 Globally Executable MHP specification(GEM 1.0.2)for package mediatargets，来构成 Java(TM) 平台，可以通过将下面所示的功能构成要素设置在该 Java(TM) 平台上而在工业上生产。

[0120] 图 6 是表示本实施形态中的应用程序执行装置的功能结构框图。应用程序执行装置由 BD 驱动器 1、应用程序管理器 2、虚拟机 3、硬盘 4、安全管理器 5 构成。

[0121] (BD 驱动器 1)

[0122] BD 驱动器 1 进行 BD-ROM 的装载 / 弹出，执行 BD-ROM 内的数据访问。在驱动器 1 中将 BD-ROM 装载 / 弹出 BD-ROM 的情况下，BD 驱动器 1 通知应用程序管理器 2 该内容。

[0123] （应用程序管理器 2）

[0124] 应用程序管理器 2 是在虚拟机 3 内的堆区内动作的系统应用程序，执行应用程序信令。所谓“应用程序信令”是指在 GEM1.0.2 规定的 MHP(Multimedia Home Platform) 中，将“服务”作为生存区间来进行应用程序的启动、执行的控制。本实施形态中的应用程序管理器 2 代替该“服务”，将 BD-ROM 中的“标题”作为生存区间，来实现应用程序的启动、执行的控制。这里，所谓“标题”是在 BD-ROM 中记录的视频・声音数据的再现单元，唯一分配应用程序管理表 (AMT)。

[0125] 每次启动应用程序时，应用程序管理器 2 判断要启动的应用程序是否正当。该判断顺序如下。若装载 BD-ROM，则确认 /BDDATA/BD.ROOT.CERTIFICATE 这样的文件的存在。在文件存在的情况下，应用程序管理器 2 从 BD-ROM 中读出该盘根证书 301，而保持在存储器上。之后，读出 Java(TM) 存档文件 302，而验证该 Java(TM) 存档文件 302 上存在的签名。若该验证正当，则应用程序管理器 2 将 BD-ROM 上存在的 Java(TM) 存档文件 302 中的类文件 401 读到虚拟机 3 内，通过在堆区上生成该类文件 401 的实例，而启动 Java(TM) 应用程序。

[0126] (虚拟机 3)

[0127] 虚拟机 3 是由将从 BD-ROM 中读出类文件的用户类装载器、将与类文件对应的实例 (instance) 作为 Java(TM) 应用程序来加以存储的堆内存、线程和 Java(TM) 堆栈构成的 Java(TM) 应用程序的执行主体。这里线程是执行 Java(TM) 应用程序中的方法的逻辑执行主体，将本地变量或操作数堆栈上存储的自变量作为操作数来加以运算，并将运算结果存储到本地变量或操作数堆栈上。基于堆栈的方法执行通过在将形成方法的字节码变换为 CPU 的本机代码 (native code) 后，发送到 CPU 中来加以执行。对于该本机代码变换，由于在本申请的主题之外，所以省略说明。在 Java(TM) 存档文件 302 内存在允许请求文件 405 的情况下，若在清单文件 402 中不存在 Java(TM) 应用程序的正确哈希值，则不执行该 Java(TM) 应用程序。为了进行该哈希值的判断，虚拟机 3 在存储器上保持表示将执行的 Java(TM) 应用程序存储在哪个 Java(TM) 存档文件 302 上的信息。通过参考该允许请求文件 405，虚拟机 3 确认应用程序管理器 2 保持的应用程序之间通信的许可，而对 Java(TM) 应用程序提供应用程序之间通信的功能。

[0128] (硬盘 4)

[0129] 硬盘 4 是通过使用来自 Java(TM)IO Package 的方法，可以加以访问的本地存储器。该本地存储器具有多个域区域。这里所谓域区域是指与各盘根证书 301 对应的目录 (图中的 R1、R2)，在这些目录的下面存储每个组织的目录 (图中的 org1，org2，org3)。组织的每个应用程序的目录 (图中的 org1/app1，org1/app2，org1/app3....) 与 MHP 中的相同。即，本地存储器中，为将 MHP 中规定的各组织的每个应用程序的目录 (图中的 org1/app1，org1/app2，org1/app3....) 配置在与根证书对应的目录 (图中的 R1、R2) 下的结构。由此，可以与 MHP 的存储方式维持置换。这里将访问本地存储器的目录结构用的文件路径中，到与根证书对应的部分 (图中的 Root/R1，Root/R2) 称作“本地存储器根”。

[0130] (安全管理器 5)

[0131] 安全管理器 5 保持表示了多个从根证书算出的哈希值和本地存储器根的组的哈希管理表，若要求从应用程序中读出 / 写入文件，则对与请求源的应用程序对应的根证书，算出哈希值，并从哈希管理表中选出与这样算出的哈希值对应的本地存储器根。将这样选出的本地存储器根嵌入到文件路径上。另外，根据 Credential 来替换与文件路径的组织 ID 对应的目录。由此，应用程序的文件路径的描述可以保持与 MHP 中规定的文件路径互相置换。

[0132] 之后，说明应用程序管理器 2、安全管理器 5 进行的具体软件的安装。应用程序管理器 2 生成如图 7 所示的程序，并通过使 CPU 加以执行，而可以安装到应用程序执行装置上。

[0133] 图 7 是表示应用程序管理器 2 进行的基于 Java(TM) 存档文件 302 内的类文件 401 的应用程序的启动顺序的流程图。应用程序管理器 2 确认在 Java(TM) 存档文件 302 中是否存在 SIG-BD.SF、SIG-BD.RSA、bd.XXXX.perm(SA01)。在一个都不存在的情况下，有可能窜改了 Java(TM) 应用程序，而不执行 Java(TM) 应用程序 (SA04)。

[0134] 在上述三个文件都存在的情况下，利用 MANIFEST.MF、SIG-BD.SF、SIG-BD.RSA，来签名验证 bd.XXXX.perm 和 Java(TM) 应用程序 (SA03)。在签名验证不成功的情况下，存储无有效的 bd.XXXX.perm，而使虚拟机 3 执行具有缺省的允许的类文件 401(SA02)。

[0135] 在签名验证成功的情况下，判断在第一数字证书链中存在的根证书和 BD.ROOT.CERTIFICATE 内的盘根证书 301 的一致性 (SA05)。在各个根证书不同的情况下，判断为 Java(TM) 存档文件 302 不正当，则不执行 Java(TM) 应用程序 (SA04)。

[0136] 在各个根证书有一致性的情况下，确认在第一数字证书链中的叶证书内是否存在组织 ID(SA06)。在不存在组织 ID 的情况下，判断为 Java(TM) 存档文件 302 不正当，而不执行 Java(TM) 应用程序 (SA04)。

[0137] 在存在组织 ID 的情况下，确认在 bd.XXXX.perm 中是否存在 Credential(SA07)。在不存在的情况下，接着到步骤 SA10。

[0138] 在存在 Credential 的情况下，验证 Credential(SA08)。后面描述验证处理的细节。在 Credential 有多个的情况下，对各 Credential 来执行该步骤。

[0139] 在 Credential 的验证中任何一次有失败的情况下 (SA09 中 No)，判断为 Java(TM) 存档文件 302 不正当，而不执行 Java(TM) 引用程序 (SA04)。

[0140] 在 Credential 的验证全部成功的情况下，或不存在 Credential 的情况下，存储 bd.XXXX.perm、组织 ID 和 MANIFEST.MF 内存在的应用程序 ID，将在 MANIFEST.MF 中记载的最先应执行的类文件 401 读入到虚拟机上，而执行作为这些的实例的 Java(TM) 应用程序 (SA10)。

[0141] 参考图 8，来说明应用程序管理器 2 验证 Credential 的流程图。图 8 是表示应用程序管理器 2 进行的 Credential 的签名验证的顺序的流程图。

[0142] 首先，开始确认接受者根证书的哈希值 503 是否与 BD.ROOT.CERTIFICATE 中的盘根证书 301 是否一致 (SY01)。在不一致的情况下，验证失败 (SY02)。

[0143] 确认接受者组织 ID504 和第一数字证书链中的叶证书上记载的组织 ID 是否一致 (SY03)。在不一致的情况下，验证失败 (SY02)。

[0144] 确认接受者应用程序 ID505 是否与 bd.XXXX.perm 内的证书之外的位置上记载的应用程序 ID 一致 (SY04)。在不一致的情况下，验证失败 (SY02)。确认 Credential 中的提供文件的名字的开头与提供者组织 ID502 是否一致 (SY05)。在不一致的情况下，验证失败 (SY02)。

[0145] 确认 SIG-BD.RSA 中的签名信息的正当性 (SY06)。在签名信息不正当的情况下，可能窜改了 Credential，验证失败 (SY02)。

[0146] 在签名信息正当的情况下，确认第二数字证书链中的根证书的哈希值和 Credential 中的提供者根证书的哈希值的一致性 (SY07)。在不一致的情况下，Credential 不正当，验证失败 (SY02)。

[0147] 确认第二数字证书链中的叶证书的组织 ID 是否与 Credential 提供者组织 ID502

一致 (SY08)。在不一致的情况下，证书不正当，验证失败 (SY02)。

[0148] 在所有的确认成功的情况下，验证成功 (SY09)。

[0149] 图 9 是在本发明的应用程序执行装置中应用程序管理器 2 保持的管理信息的一例。以表格的形式管理盘根证书 301、所执行的 Java(TM) 存档文件 302 的“Jar 文件名”、“组织 ID”、“应用程序 ID”、“应用程序之间通信”和“Credential”。

[0150] 图 10 是在本发明的应用程序执行装置中，应用程序管理器 2 保持的管理信息中的 Credential 表格的一行的一例。Credential 表格包括“提供者根证书的哈希值”501、“提供者组织 ID”502 和“提供者文件列表”506。在该“提供者文件列表”506 中记载了提供文件名和提供者访问方法 508。

[0151] 在向 Java(TM) 存档文件 302 提供了允许的情况下，虚拟机 3 在 Java(TM) 应用程序的执行前进行下面的处理。虚拟机 3 确认在 MANIFEST.MF 中的 Java(TM) 应用程序的哈希值和 Java(TM) 应用程序的实际哈希是否一致，在不一致的情况下，不加以执行。

[0152] Java(TM) 应用程序通过虚拟机 3 来加以执行。参考图 11，说明在典型的 Java(TM) 应用程序想要利用本地存储器的情况下所进行的处理顺序。

[0153] 图 11 是表示每次在 Java(TM) 应用程序利用硬盘 4 时的处理顺序的流程图。

[0154] Java(TM) 应用程序取得虚拟本地存储器根 (SC01)。这里所谓虚拟本地存储器根是指以 MHP 中的形式来指定作为访问目标的文件的文件路径。以 /Root 或 /Storage/Root 的名字来加以表现。该虚拟本地存储器根不过是想要与 MHP 中的文件路径的形式相互置换，在硬盘 4 中的文件访问时，将该本地存储器名变换为按每个根证书决定的本地存储器根。

[0155] 接着，Java(TM) 应用程序取得组织 ID(SC02)。组织 ID 例如是“4”的数字。

[0156] Java(TM) 应用程序组合本地存储器根的名字和组织 ID，指定想要读写的文件的名字，来进行对文件的输入输出 (SC03)。例如，在本地存储器根的名字是“/persistent/0003”，组织 ID 是“7”，想要读写的文件的相对路径是“8/scores.txt”的情况下，可以通过“/persistent/0003/7/8/scores.txt”的全路径来指定文件。

[0157] Java(TM) 应用程序为了进行如图 11 的处理，安全管理器 5 对 Java(TM) 应用程序提供下面的函数调用。

[0158] （ア）本地存储器根的取得

[0159] （イ）组织 ID 的取得

[0160] （ウ）从文件的读出

[0161] （エ）向文件的写入

[0162] 参考图 12，来说明 Java(TM) 应用程序进行的“本地存储器根的取得”函数调用的处理的流程图。

[0163] 首先，安全管理器 5 确认调用源的 Java(TM) 应用程序上是否存在 bd.XXXX.perm(SD01)。是否存在 bd.XXXX.perm 可以从虚拟机 3 中取得存储了以 Java(TM) 应用程序为实例的类文件 401 的 Java(TM) 存档文件 302，而从应用程序管理器 2 中取得。在不存在 bd.XXXX.perm 的情况下，拒绝“本地存储器根的取得”(SD02)。

[0164] 在存在 bd.XXXX.perm 的情况下，从应用程序管理器 2 中取得盘根证书 301 的哈希值 (SD03)。安全管理器 5 确认哈希管理表格（后述），来确认是否已经登记了盘根证书 301

的哈希值 (SD04)。在登记了的情况下，将与盘根证书 301 的哈希值对应的本地存储器根返回到 Java(TM) 应用程序中 (SD05)。

[0165] 在哈希管理表格上没有登记盘根证书 301 的哈希值的情况下，安全管理器 5 将新的条目追加到哈希管理表上 (SD06)。在该条目上记载盘根证书 301 的哈希值和在表格内唯一的本地存储器根。将在哈希管理表上新登记的条目的本地存储器根返回到 Java(TM) 应用程序 (SD07)。

[0166] 参考图 13，表示安全管理器 5 保持的哈希管理表的一例。哈希管理表中存在根证书的哈希值 1301 和本地存储器根 1302。该图中的本地存储器根的“/0001”，“0003”分别是指图 6 的目录名 /R1，/R2。这些本地存储器根 1302 与根证书的哈希值 1301 一一对应。因此，对具有同一盘根证书 301 的 BD 盘中的作为 Java(TM) 存档文件 302 中的类文件 401 的实例的 Java(TM) 应用程序返回同一本地存储器根 1302，对具有不同的盘根证书 301 的 BD-ROM 中的作为 Java(TM) 存档文件 302 中的类文件 401 的实例的 Java(TM) 应用程序返回不同的本地存储器根 1302。

[0167] 参考图 14，来说明“组织 ID 的取得”函数调用的流程图。

[0168] 首先，安全管理器 5 确认调用源的 Java(TM) 应用程序中是否存在 bd.XXXX.perm(SF01)。在不存在 bd.XXXX.perm 的情况下，由于不能确认组织 ID，所以拒绝“组织 ID 的取得”(SF02)。在存在 bd.XXXX.perm 的情况下，从应用程序管理器 2 中取得与 Java(TM) 应用程序对应的组织 ID，而返回到 Java(TM) 应用程序 (SF03)。

[0169] “文件读出”函数调用将想要读出的文件名的全路径作为参数从 Java(TM) 应用程序传送到安全管理器 5 中。该全路径的形式为组织 ID/appID，设应用程序以与访问 MHP 中的本地寄存器内的文件相同的顺序，来访问应用程序执行装置内的文件。

[0170] 在文件访问成功的情况下，安全管理器 5 将数据返回到 Java(TM) 应用程序。参考图 15 和图 16，来说明“文件读出”函数调用的流程图。

[0171] 首先，安全管理器 5 确认调用源的 Java(TM) 应用程序中是否存在 bd.XXXX.perm(SH01)。在不存在 bd.XXXX.perm 的情况下，拒绝“文件读出”(SH02)。

[0172] 在存在 bd.XXXX.perm 的情况下，从应用程序管理器 2 取得盘根证书 301 的哈希值 (SH03)。

[0173] 安全管理器确认哈希管理表格，取得与盘根证书 301 的哈希值对应的本地存储器根 1302(SH04)。

[0174] 接着，判断指定文件名的文件的开头根名是否与虚拟本地存储器根的名字一致 (SH05)。其检查是否通过依赖于装置的形式（这里为与 MHP 可替换的形式）的文件路径来指定访问目标文件，在一致的情况下，将根名变换为作为与盘根证书对应的形式的本地存储器根 (SH13)。在不一致的情况下，拒绝进行访问 (SH02)。

[0175] 这里，在变换前后，全路径如下那样。

[0176] 应用程序指定的全路径：

[0177] / 虚拟本地存储器根 / 指定组织 ID/ 指定路径

[0178] SH05 进行的变换后的路径：

[0179] / 本地存储器根 / 指定组织 ID/ 指定路径

[0180] 之后，安全管理器 5 分解文件名的全路径 (SH06)。由于文件名的全路径利用‘本

地存储器根 1302+“/”+指定组织 ID+“/”+指定路径’的形式，所以可以将文件名的全路径分解为本地存储器根 1302、指定组织 ID 和指定路径。在不能分解的情况下，拒绝读出(SH02)。

[0181] 安全管理器 5 从应用程序管理器 2 中取得调用源的 Java(TM) 应用程序的组织 ID(SH07)。并且，尝试是否可从应用程序管理器 2 中取得指定组织 ID 与提供者组织 ID502 一致的 Credential(SH10)。在不能取得 Credential 的情况下，拒绝访问 (SH02)。

[0182] 在取得了 Credential 的情况下，确认来自 Java(TM) 应用程序的指定路径是否存在于 Credential 的提供文件名 506 中，是否通过提供访问方法 507 来确认允许读出(SH11)。在允许的情况下，安全管理器 5 取得 Credential 内的提供者根证书的哈希值 501(SH12)。

[0183] 在不允许的情况下，安全管理器 5 确认指定组织 ID 和 Java(TM) 应用程序的组织 ID 是否一致 (SH08)。在一致的情况下，根据 Java(TM) 应用程序指定的全路径，从硬盘中读出文件而返回到应用程序中 (SH09)。在不一致的情况下，拒绝读出 (SH02)。

[0184] 安全管理器 5 确认哈希管理表格，来确认是否已经登记了提供者根证书的哈希值 501(SH13)。在登记了的情况下，取得与提供者根证书的哈希值 501 对应的本地存储器根 1302，而作为提供者本地存储器根来加以确定 (SH14)。

[0185] 在哈希管理表格上没有登记提供者根证书的哈希值 501 的情况下，安全管理器 5 在哈希管理表格上追加新的条目 (SH15)。在该条目上记载提供者根证书的哈希值 501 和表格内唯一的本地存储器根 1302。

[0186] 将在哈希管理表上新登记的行的本地存储器根 1302 作为提供者本地存储器根来加以确定 (SH16)。

[0187] 在确定了提供者本地存储器根后，安全管理器 5 将文件名的全路径中的本地存储器根 1302 替换为提供者本地存储器根 (SH17)。

[0188] 安全管理器 5 通过硬盘 4 读出替换后的文件名的全路径的文件，而返回到 Java(TM) 应用程序 (SH18)。

[0189] 以上是关于文件读出的说明。接着，说明文件写入。

[0190] 在“文件写入”函数调出时，将想写入的文件名的全路径和想写入的数据作为参数从 Java(TM) 应用程序传到安全管理器 5。在成功的情况下，安全管理器 5 将数据写入到文件中。参考图 17 和图 18，来说明“文件写入”函数调用的流程图。

[0191] 首先，安全管理器 5 确认调用源的 Java(TM) 应用程序上是否存在 bd.XXXX.perm(SI01)。在不存在 bd.XXXX.perm 的情况下，拒绝“文件写入”(SI02)。

[0192] 在存在 bd.XXXX.perm 的情况下，通过应用程序管理器 2 取得盘根证书 301 的哈希值 (SI03)。

[0193] 安全管理器确认哈希管理表，并取得与盘根证书 301 的哈希值对应的本地存储器根 1302(SI04)。

[0194] 并且，判断指定文件名的全路径的开头根名是否与虚拟本地存储器根的名字一致(SI05)。在一致的情况下，将根名变换为本地存储器根 (SI13)。在不一致的情况下，拒绝访问 (SI02)。

[0195] 这里，在变换前后，全路径如下那样。

[0196] 应用程序指定的全路径：

[0197] / 虚拟本地存储器根 / 指定组织 ID/ 指定路径

[0198] 基于 SH05 的变换后的路径：

[0199] / 本地存储器根 / 指定组织 ID/ 指定路径

[0200] 之后，安全管理器 5 分解文件名的全路径 (SI06)。由于文件名的全路径使用‘本地存储器根 1302+“/”+ 指定组织 ID+“/”+ 指定路径’的形式，所以可文件名的全路径可以分解为本地存储器根 1302、指定组织 ID 和指定路径。在不能分解的情况下，拒绝写入 (SI02)。

[0201] 安全管理器 5 取得调用源的 Java(TM) 应用程序的组织 ID(SI07)。并且，尝试能否从应用程序管理器 2 中取得指定组织 ID 与提供者组织 ID502 一致的 Credential(SI10)。在不存在的情况下，拒绝写入 (SI02)。

[0202] 在取得了的情况下，来自 Java(TM) 应用程序的指定路径作为 Credential 的提供文件名 506 存在，并通过提供访问方法 507 来确认是否允许写入 (SI11)。在不允许的情况下，安全管理器 5 确认指定组织 ID 和 Java(TM) 应用程序的组织 ID 是否一致 (SI08)。在一致的情况下，根据 Java(TM) 应用程序指定的全路径，将数据写入到文件中 (SI09)。在不一致的情况下，拒绝写入 (SI02)。

[0203] 在确认允许写入的情况下，安全管理器 5 取得 Credential 内的提供者根证书的哈希值 501(SI12)。

[0204] 安全管理器 5 确认哈希管理表，来确认是否已经登记了提供者根证书的哈希值 501(SI13)。在登记了的情况下，取得与提供者根证书的哈希值 501 对应的本地存储器根 1302，而作为提供者本地存储器根来加以确定 (SI14)。

[0205] 在哈希管理表上没有登记提供者根证书的哈希值 501 的情况下，安全管理器 5 将新行追加到哈希管理表上 (SI15)。该行上记载了提供者根证书的哈希值 501 和表格内唯一的本地存储器根 1302。

[0206] 将在哈希管理表上新登记的行的本地存储器根 1302 作为提供者本地存储器根来加以确定 (SI16)。

[0207] 在确定了提供者本地存储器根后，安全管理器 5 将文件名的全路径中的本地存储器根 1302 替换为提供者本地存储器根 (SI17)。

[0208] 安全管理器 5 写入替换后的文件名的全路径的文件 (SI18)。

[0209] 如上所述，根据本实施形态，由于在本地存储器 4 的内部存在分别分配给根证书的哈希值的目录，所以若在这些目录下按每个组织来生成每个应用程序的领域，则在世界范围中，组织 ID 可以不唯一。由于在“域区域”这样封闭的世界中，可区别多个组织就足够了，所以在世界范围中组织 ID 不需要是唯一的值，不需要基于第三人机关的管理。

[0210] 由于根证书不分配给装置主体，而分配给存储单元内的域区域，所以只要将 BD-ROM 装到应用程序执行装置中，必然保障了动作。尤其，并非没有可能暴露盘根证书，但是在这种情况下，使得不能使用该 BD-ROM，或可以仅更新对该 BD-ROM 的盘根证书就可以了，由于由其他 BD-ROM 供给的应用程序可以如现有技术那样，使用盘根证书，所以可以实现可靠的动作保障。

[0211] 这样，由于不需要这种世界范围内的组织 ID 的管理，且将现有的应用程序的彼此

保障维持在高水平上，所以本发明的应用程序执行装置对执行与视频作品有关的应用程序的应用程序执行装置的世界标准化有很大贡献。

[0212] （补充）

[0213] 以上，说明了在本申请的申请时刻，申请人可以知道的最佳实施形态，可以对下面所示的技术论点来提供进一步的改良和改变实施。如各实施形态所示那样来实施，或是否实施这些改进·改变都是任意的，注意依赖于实施人的主观意愿。

[0214] (Java(TM) 存档文件 302 的选择）

[0215] 在 BD-ROM 中还存在有视频等的数据的情况下，考虑选择通过视频再现中的事件（章节 2 的再现开始时）指定的 Java(TM) 存档文件 302 和与用户屏幕上的选择对应的 Java(TM) 存档文件 302。

[0216] 有时数字证书链由一个证书构成。这时，根证书、叶证书都是同一证书。

[0217] （许可的种类）

[0218] 在是具有更丰富的功能的应用程序执行装置的情况下，也可在 bd.XXXX.perm 中包含其他种类的许可。也可在 bd.XXXX.perm 中包含多个数字信用证书 312。

[0219] （组织 ID)

[0220] 根据实施形态来考虑将组织 ID 记载在 BD-ROM 上不同的文件上。这时，也可不需要拒绝“组织 ID 的取得”，而返回通过其他方法确定的组织 ID。

[0221] （文件的读出 / 写入）

[0222] 图 15、图 16、图 17、图 18 中以文件的读出为中心进行了说明，但是同样还可从 Java(TM) 应用程序进行直接访问。在直接访问的情况下，还考虑在文件名的全路径上不存在指定路径的情况。

[0223] 图 15 和图 16 中，作为读取文件整体的函数调用进行了说明，但是同样还可部分取得文件等，来进行典型的文件访问。

[0224] 图 11 所示的控制流程是一例，根据 Java(TM) 应用程序的创建而大大不同。

[0225] （与其他方式的兼用）

[0226] 本方式可以与其他文件访问许可方式（例如在 UNIX（注册商标）上广泛使用的用户·组·全球访问模块）兼用。例如，也可兼用本方式和第二方式来决定下面的优先级。

[0227] （ア）在步骤 SH06 中不能分解为全路径的情况下，或步骤 SH10 中不存在数字信用证书的情况下，利用第二方式的访问控制。

[0228] （イ）在步骤 SH09 中使用第二方式的访问控制。

[0229] （ウ）对于其他，本方式优先。

[0230] （哈希值）

[0231] 所谓本实施形态中的哈希值是指利用了 SHA-1 或 MD5 等的安全哈希函数的结果值。安全哈希函数具有特征 :实质上不可能发现具有相同哈希值的数据。

[0232] （根证书的哈希值）

[0233] 所谓本实施形态中的根证书的哈希值，可以不需要必然从根证书整体的数据中算出，而至少仅从根证书中包含的公钥的数据中算出。盘创建者可以明示选择用于 MANIFEST.MF、SIG-BD.SF、SIG-BD.RSA 中存储的哈希值的计算的安全哈希函数。

[0234] 本实施形态中，将 bd.XXXX.perm 中的数字信用证书上固定了用于提供者根证书

的哈希值 501 和接受者根证书的哈希值 503 的计算的安全哈希函数的情况作为连接对方，但是也可在 bd.XXXX.perm 中的数字信用证明中明示用于提供者根证书的哈希值 501 和接受者根证书的哈希值 503 的计算的安全哈希函数。

[0235] （根证书的比较）

[0236] 步骤 SA05 中的根证书的比较可以进行根证书是否相同的比较，根证书中包含的公钥是否相同的比较。作为其他方式，也可忽略数字签名文件中的第一证书（根证书），确认是否通过盘根证书 301 签名了接着根证书的第二证书。由于不管使用哪种方式，盘根证书 301 都保障了数字签名文件中的第二证书，所以安全观点的效果相同。

[0237] 步骤 SA05 的比较的主要目的是防止恶意使用应用程序之间的通信的攻击。认为恶意使用应用程序之间的通信的攻击试着使用如下这样产生的攻击用 BD-ROM。

[0238] 从通过攻击对象的盘创建者创建的正当的 BD-ROM 中读取通过数字证书签名的攻击对象的 Java(TM) 应用程序 302。

[0239] 生成进行攻击用的 Java(TM) 存档文件 302，并通过数字证书来加以签名。

[0240] 将攻击对象的 Java(TM) 存档文件 302 和攻击用的 Java(TM) 存档文件 302 记录在攻击用 BD-ROM 上。

[0241] 攻击用的 Java(TM) 存档文件 302 和攻击对象的 Java(TM) 存档文件 302 同时通过数字证书来加以签名，但是都使用不同的根证书。若在应用程序执行装置中，对两个 Java(TM) 存档文件 302 中的 Java(TM) 应用程序许可应用程序之间通信，则攻击用的 Java(TM) 存档文件 302 可以对攻击对象的 Java(TM) 存档文件 302 进行不正当的应用程序之间的通信，攻击对象的 Java(TM) 存档文件 302 要对自身使用的存储区域进行攻击对象的盘创建者不能预期的动作。

[0242] 为了防止上述攻击，需要在步骤 SA05 中比较根证书。另外，也可代替步骤 SA05，防止利用不同的根证书的 Java(TM) 应用程序彼此的应用程序之间的通信。这时，一个 BD-ROM 上也可具有多个盘根证书 301。

[0243] （本地存储器名的名字取得函数）

[0244] 实施形态中，在本地存储器名的名字取得函数的步骤 SC01 中，将作为与 MHP 可替换的形式的文件路径的虚拟本地存储器名暂时返回到应用程序后，在 SH05、SI05 中，变换为应用程序执行装置中的本地存储器名，但是也可在本地存储器名的名字取得函数的步骤 SC01 中，直接将在哈希管理表上描述的本地存储器名直接返回到 Java(TM) 应用程序，而省去虚拟本地存储器名向本地存储器名的变换。

[0245] （标题）

[0246] 在制造应用程序执行装置来作为 BD-ROM 再现装置的情况下，最好将根据 BD-ROM 的装载或用户操作、装置的状态来选择标题的“模块管理器”设置在应用程序执行装置上。BD-ROM 再现装置内的解码器根据基于该“模块管理器”的标题选择，来进行基于播放列表信息的 AVClip 的再现。

[0247] 应用程序管理器 2 在“模块管理器”进行了标题的选择时，使用与前一标题对应的 AMT 和与当前标题对应的 AMT 来执行信令。该信令记载在与前一标题对应的 AMT 上，但是进行使没有在与当前标题对应的 AMT 上记载的应用程序的动作终止，不被记载在与前一标题对应的 AMT 上，使与当前标题对应的 AMT 上所记载的应用程序的动作开始的控制。并且，每

次进行该应用程序信令时，最好进行使用上述的盘根证书的验证。

[0248] (BD-BOX)

[0249] 在将长篇视频作品或一组视频作品记录在多个 BD-ROM 上，来构成所谓的 BD-BOX 时，最好将同一盘根证书 301 分配给这些 BD-ROM。这样，在将同一盘根证书 301 分配给多个 BD-ROM 的情况下，可通过应用程序，在盘交换前后进行动作。这样，将在盘交换前后动作的应用程序称作“盘无边界应用程序”。这里应用程序管理器 2 最好在进行了 BD-ROM 的交换的情况下，读出新装入的 BD-ROM 的盘根证书 301，并进行确认与定义该盘无边界应用程序的 Java(TM) 存档文件 302 内的盘根证书 301 的一致性的处理。并且，若有一致性，则继续进行盘无边界应用程序，若不具有一致性，则强制终止盘无边界应用程序。由此，在 BD-ROM 的交换前后，可以仅使正当的应用程序动作。

[0250] (Credential)

[0251] • Credential 最好在多个 XML 文件中取出用特定的标志包围的部分，通过结合这些部分来构成。

[0252] • 也可将通过提供者组织的叶证书（的公钥）签名了 Credential 后的值作为 Credential 的签名信息，记载在 bd.XXXX.perm 中。

[0253] • 在进行“权限的提供”的情况下，bd.XXXX.perm 可以是多个，但是在有多个的情况下，bd.XXXX.perm 中，最好描述 SIG-BD.SF 的哪个叶证书用于与哪个 Credential 的对照的信息。

[0254] • 作为 Credential 的生成，可以在 bd.XXXX.perm 上写入提供者文件名，并从其他文件中写入的值中算出 Credential 的实体。

[0255] • 并且，全部组合这些，最好在 bd.XXXX.perm 中得到了提供者文件、特定叶证书的信息、签名信息。

[0256] （本地存储器）

[0257] 本实施形态中本地存储器为装置嵌入型的硬盘，若是安全的记录介质，也可采用可移动的记录介质。例如，也可采用 SD 存储卡。

[0258] （应安装的包）

[0259] 每次实施应用程序执行装置时，最好将下面的 BD-J Extention 安装到应用程序执行装置上。BD-J Extention 包含为将超过 GEM[1.0.2] 的功能提供给 Java(TM) 平台而具体化的各种包。由 BD-J Extention 提供的包有如下几个。

[0260] • org.bluray.media

[0261] 该包提供应追加到 Java(TM)Media FrameWork 的特殊功能。将对于角度、声音、字幕的选择的控制追加到该包上。

[0262] • org.bluray.ti

[0263] 该包包含将 GEM[1.0.2] 中的“服务”映射到“标题”上来加以动作的 API、从 BD-ROM 中询问标题信息的结构和选择新标题的机构。

[0264] • org.bluray.application

[0265] 该包包含管理应用程序的生存区间用的 API。另外，包含询问每次执行应用程序时的信令所需的信息的 API。

[0266] • org.bluray.ui

[0267] 该包包含定义 BD-ROM 中特殊化的关键事件用的常数，而实现与视频再现的同步的类。

[0268] • org.bluray.vfs

[0269] 该包由于不局限于数据的位置，来无缝再现数据，所以提供绑定在 BD-ROM 上记录的内容 (on-disc 内容) 和没有在 BD-ROM 上记录的本地存储器上的内容 (off-disc 内容) (Binding Scheme)。

[0270] 所谓 Binding Scheme，是指使 BD-ROM 上的内容 (AVClip、字幕、BD-J 应用程序) 和本地存储器上的关联内容相关联。该 Binding Scheme 实现无缝再现，而不限于内容的位置。

[0271] (本地存储器的访问)

[0272] 例如，确认在文件路径“/Persistent/1/1/streams”上是否存在希望的文件，可以通过使用 java(TM).io 的 exits() 方法来进行。下面表示希望的文件是 0.m2ts 的情况下的 java(TM).io 的 exits() 方法的使用例。

[0273] 例 :

[0274] new java(TM).io.File(”/Persistent/1/1/streams/0.m2ts”).exists() ;

[0275] (许可请求文件)

[0276] 许可请求文件也可决定是否允许下面的功能。

[0277] • 网络连接的使用

[0278] • BD-ROM 的访问

[0279] • BD-ROM 中的其他标题的选择

[0280] • 其他平台的执行控制

[0281] (视频 • 声音)

[0282] 也可在图 1(a) 所示的 BD-ROM 的目录结构中，在 ROOT 目录下设置 BDMV 路径，在该目录下记录作为 MPEG2-TS 形式的 AV 流的 AVClip 和规定该再现路径的播放列表信息。并且，也可描述要进行通过播放列表信息的再现的 Java(TM) 应用程序。

[0283] 在播放列表信息存储在 00001.mpls 的文件中的情况下，Java(TM) 应用程序根据 JMF 库，来生成 JMF 播放实例。JMF A”BD ://00001.mpls”是命令虚拟机进行再现 PL 的播放实例的生成的方法。A.play 是对 JMF 播放实例命令再现的方法。

[0284] (BD-ROM 内容)

[0285] 设 BD-ROM 上所记录的应用程序构成视频作品，但是若不是在本地存储器上安装来加以使用的应用程序，而是以在 BD-ROM 上记录的状态下来使用的应用程序，则可以构成除此之外的应用程序。例如，可以是构成游戏软件的应用程序。本实施形态中作为盘介质，选择 BD-ROM 为题材，但是若是可移动体，且没有进行著作权保护的记录介质，则可以采用其他记录介质。

[0286] (Virtual Package :虚拟包)

[0287] 也可使安全管理器 5 进行生成虚拟包这样的处理。所谓虚拟包是指动态组合在 BD-ROM 等的只读型记录介质上记录的数字流和在硬盘等的可重写型的记录介质上记录的数字流，来构筑虚拟的包，来实现只读型记录介质的内容扩展的技术。这里在 BD-ROM 上记录的数字流构成视频作品的正编，在硬盘上记录的数字流构成视频作品的续编的情况下，

通过构筑上述的虚拟包，可以将 BD-ROM 上的正编和硬盘上的续编作为一个长篇的视频作品来加以处理，从而供给再现。

[0288] 其可以通过安全管理器 5 生成虚拟包信息来进行。所谓虚拟包信息是指扩展了 BD-ROM 中的卷管理信息后的信息。这里卷信息是规定在某个记录介质上存在的目录文件结构的信息，包括对目录的目录管理信息和对文件的文件管理信息。所谓虚拟包信息是指通过在表示 BD-ROM 的目录文件结构的卷管理信息上追加新的文件管理信息，来实现 BD-ROM 中的目录文件结构的扩展。通过该虚拟包信息的生成，应用程序以与访问 BD-ROM 相同的感觉，来访问本地存储器中的每个盘根证书的域区域的每个组织的区域。

[0289] （控制顺序的实现）

[0290] 由于各实施形态中引用流程图来说明的控制顺序或基于功能的构成要素的控制顺序使用硬件资源来具体实现，所以满足了称作利用了自然规律的技术思想的创建的“作为程序发明”的成立要件。

[0291] • 本发明的程序的生产形态

[0292] 本发明的程序是计算机可执行的执行形式的程序（对象程序），由使计算机执行的一个以上的程序码来构成实施形态所示的流程图的各步骤或功能构成要素的各个顺序。这里程序码有如处理器的本机代码和 JAVA 字节码那样的各种种类。基于程序码的各步骤的实现有各种形态。在可利用外部函数，来实现各步骤的情况下，调用该外部函数的调用语句是程序码。另外，有时实现一个步骤这种程序码也归属于分开的对象程序。在限制了命令种类的 RISC 处理器中，通过组合算术运算命令或逻辑运算命令、分支命令等，来实现流程图的各步骤。

[0293] 本发明的程序可以如下这样来生成。首先，软件开发者使用编程语言，来描述实现各流程图或功能结构要素的源程序。每次在该描述时，软件开发者根据编程语言的规则，使用类结构体或变量、排列变量、外部函数的调用，来描述具体实现各流程图或功能构成要素的源程序。

[0294] 将所描述的源程序作为文件提供给编译器。编译器翻译这些源程序来生成对象程序。

[0295] 基于编译器的翻译由语法分析、最佳化、资源分配、码生成的过程构成。在语法分析中，进行源程序的字句分析、语法分析和含义分析，并将源程序变换为中间程序。最佳化中，对中间程序进行基本块化、控制流程分析和数据流分析这样的操作。在资源分配中，为了实现对作为目标的处理器的命令组的适用，分配给将中间程序中的变量作为目标的处理器的处理器具有的寄存器或存储器。在码生成中，将中间程序内的各中间命令变换为程序码后，得到对象程序。

[0296] 若生成了对象程序，则程序对其启动链接程序。链接程序将这些对象程序或相关联的库程序分配给存储器空间，将这些结合为一个，来生成装载模块。这样生成的装载模块以基于计算机的读出为前提，使计算机来执行各流程图所示的处理顺序和功能构成要素的处理顺序。可以经过以上的处理，来作成本发明的程序。

[0297] 本发明的程序的使用形态

[0298] 本发明的程序可以如下这样来使用。

[0299] (i) 作为嵌入程序使用

[0300] 在本发明的程序作为嵌入程序来使用的情况下，将程序中的装载模块与基本输入输出程序 (BIOS) 和各种中间设备（操作系统）一起写入到命令 ROM 中。通过将这种命令 ROM 嵌入到控制部中，使 CPU 加以执行，而可以将本发明的程序作为应用程序执行装置的控制程序来使用。

[0301] (ii) 作为应用程序的使用

[0302] 在应用程序执行装置是硬盘内置模块的情况下，将基本输入输出程序 (BIOS) 嵌入到命令 ROM 中，将各种中间件（操作系统）预先安装到硬盘中。从硬盘将启动系统用的导入 ROM 设置在应用程序执行装置上。

[0303] 这时，仅装载模块通过可移动型的记录介质和网络，供给应用程序执行装置，并作为一个应用程序安装到硬盘上。由此，应用程序执行装置在进行基于引导 (boot)ROM 的引导程序 (bootstrap)，并启动操作系统后，作为一个应用程序，使 CPU 执行该应用程序，并使用本发明的程序。

[0304] 由于硬盘模块的应用程序执行装置中，要将本发明的程序作为一个应用程序使用，所以本发明的程序可单个转让、销售、或通过网络来提供。

[0305] （应用程序管理器 2 ～安全管理器 4)

[0306] 应用程序管理器 2 ～安全管理器 4 可以作为一个系统 LSI 来实现。

[0307] 所谓系统 LSI 是指在高密度基板上安装裸芯片后，进行封装的 LSI。使系统 LSI 包含将多个裸芯片安装在高密度基板上，通过进行封装，使多个裸芯片具有好像一个 LSI 这样的外形结构的这样的芯片（这种系统 LSI 称作多芯片模块）。

[0308] 这里若着眼于封装的种类，则系统 LSI 有 QFP（四侧引脚扁平封装）和 PGA（针脚栅格阵列）这种类别。QFP 是在封装的四个侧面上安装了针脚的系统 LSI。PGA 是在底面整体上安装了多个针脚的系统 LSI。

[0309] 这些针脚承担作为与其他电路的接口的任务。由于系统 LSI 中的针脚存在这种接口的任务，所以通过将其他电路连接到系统 LSI 中的这些针脚上，系统 LSI 实现了作为应用程序执行装置的核心的任务。

[0310] 在系统 LSI 上封装的裸芯片包括“前端部”、“后端部”和“数字处理部”。“前端部”是将模拟信号进行数字化后的部分，“后端部”是模拟处理数字处理的结果，即所得到的数据，来加以输出的部分。

[0311] 各实施形态中作为内部构成图表示的各构成要素安装在该数字处理部内。

[0312] 如之前作为“嵌入程序的使用”中所描述的，在命令 ROM 中写入了作为程序的装载模块、基本输入输出程序 (BIOS) 和各种中间件（操作系统）。本实施形态中，特别创建的是作为该程序的装载模块的部分，所以通过将存储了作为程序的装载模块的命令 ROM 作为裸芯片来封装，可以由本发明的系统 LSI 来生产。

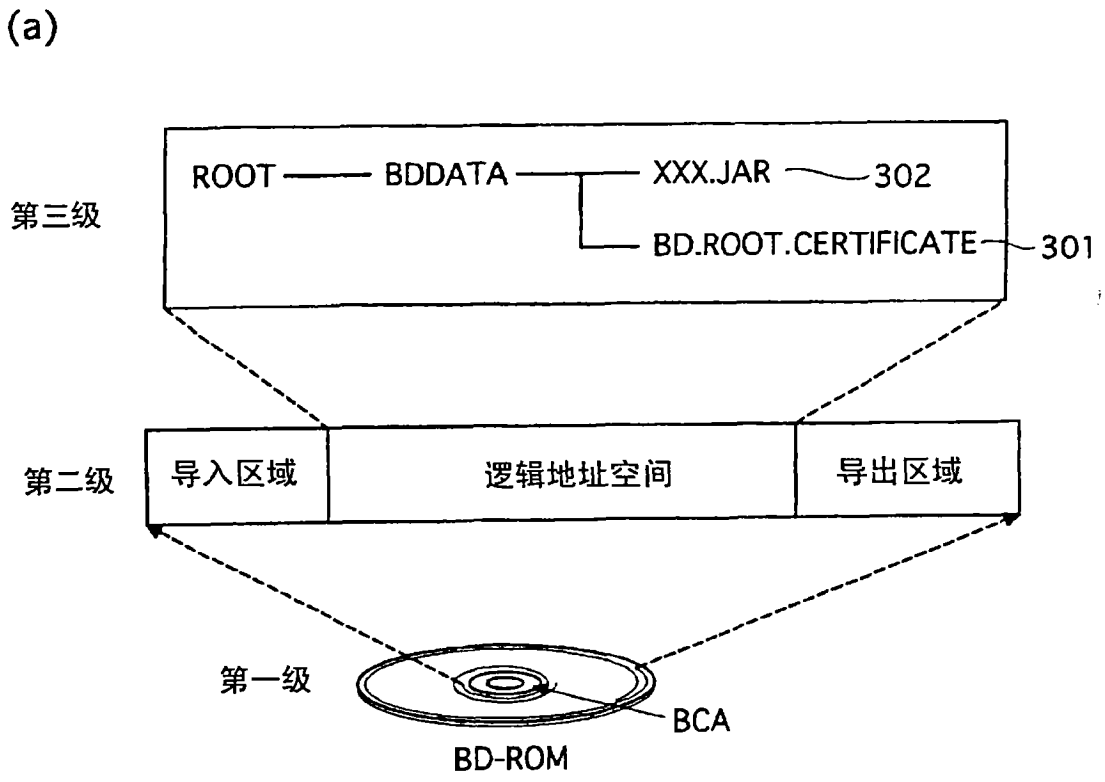
[0313] 对于具体的安装，最好使用 SoC 安装或 SiP 安装。所谓 SoC(System onchip) 安装是指将多个电路烧结在一个芯片上的技术。所谓 SiP(System inPackage) 安装是指用树脂等将多个芯片变为一个封装的技术。经过以上的过程，本发明的系统 LSI 可以以各实施形态所示的应用程序执行装置的内部构成图为基础来生成。

[0314] 另外，上述这样生成的集成电路因集成度的不同，也称作 IC、LSI、超级 LSI 和顶级 LSI。

[0315] 进一步，各记录应用程序执行装置的构成要素的一部分或全部可作为一个芯片构成。集成电路并不限于上述的 SoC 安装、SiP 安装，也可由专用电路或通用处理器来实现。考虑在LSI制造后，使用可进行编程的FPGA(Field Programable Gate Array)、或可重新构成 LSI 内部的电路单元的连接和设置的可重构处理器。进一步，若因半导体技术的进步或所派生的技术，替换 LSI 的集成电路的技术发现，则当然也可使用该技术来进行功能块的集成电路化。例如，还可以适用于生物技术。

[0316] 产业上的可利用性

[0317] 本发明的应用程序执行装置在上述实施形态中公开了内部结构，可以明白可根据该内部结构来进行生产，所以在资质上可以在工业上使用。由此，本发明的应用程序执行装置具有在产业上利用的可能性。


(a)
第三级
ROOT
BDDATA
XXX.JAR
302
BD.ROOT.CERTIFICATE
301
第二级
导入区域
逻辑地址空间
导出区域
第一级
BCA
BD-ROM

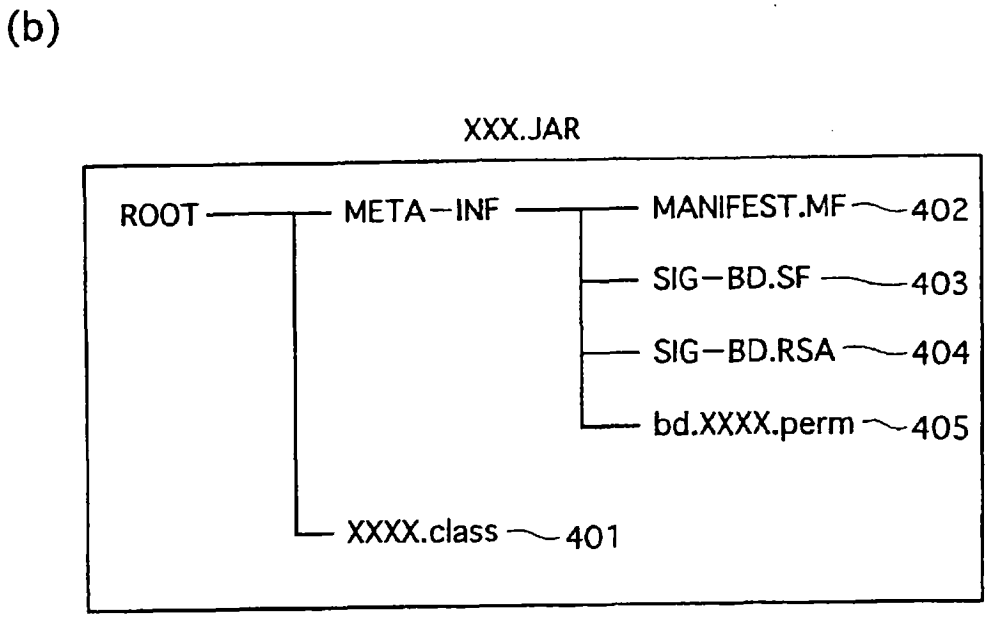

(b)
XXX.JAR
ROOT
META-INF
MANIFEST.MF
402
SIG-BD.SF
403
SIG-BD.RSA
404
bd.XXXX.perm
405
XXXX.class
401

图 1

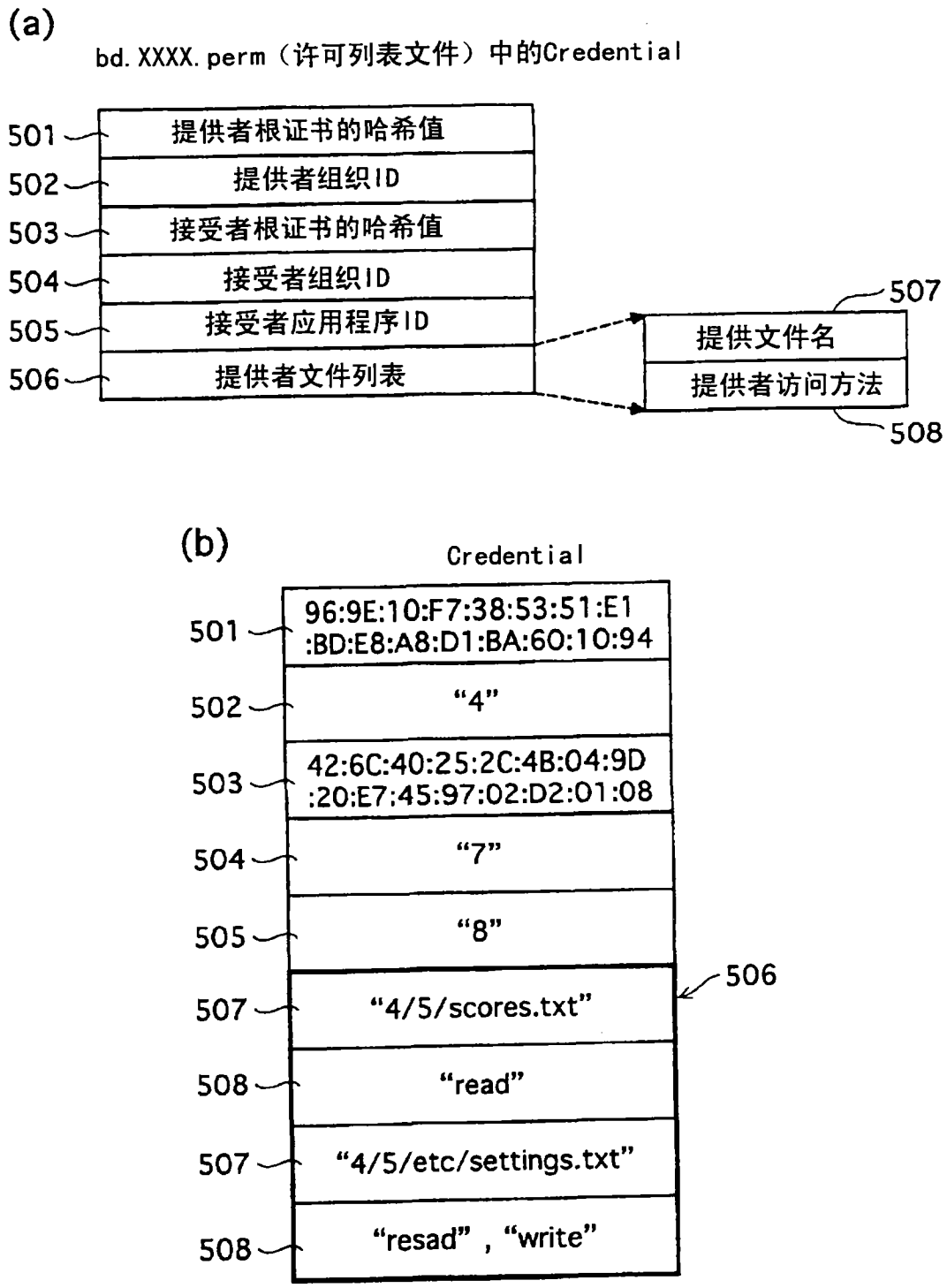

(a)
bd.XXXX.perm（许可列表文件）中的Credential
501
提供者根证书的哈希值
502
提供者组织ID
503
接受者根证书的哈希值
504
接受者组织ID
505
接受者应用程序ID
506
提供者文件列表
507
提供文件名
提供者访问方法
508
(b)
Credential
501
96:9E:10:F7:38:53:51:E1
:BD:E8:A8:D1:BA:60:10:94
502
“4”
503
42:6C:40:25:2C:4B:04:9D
:20:E7:45:97:02:D2:01:08
504
“7”
505
“8”
506
507
“4/5/scores.txt”
508
“read”
507
“4/5/etc/settings.txt”
508
“resad”，“write”

图 2

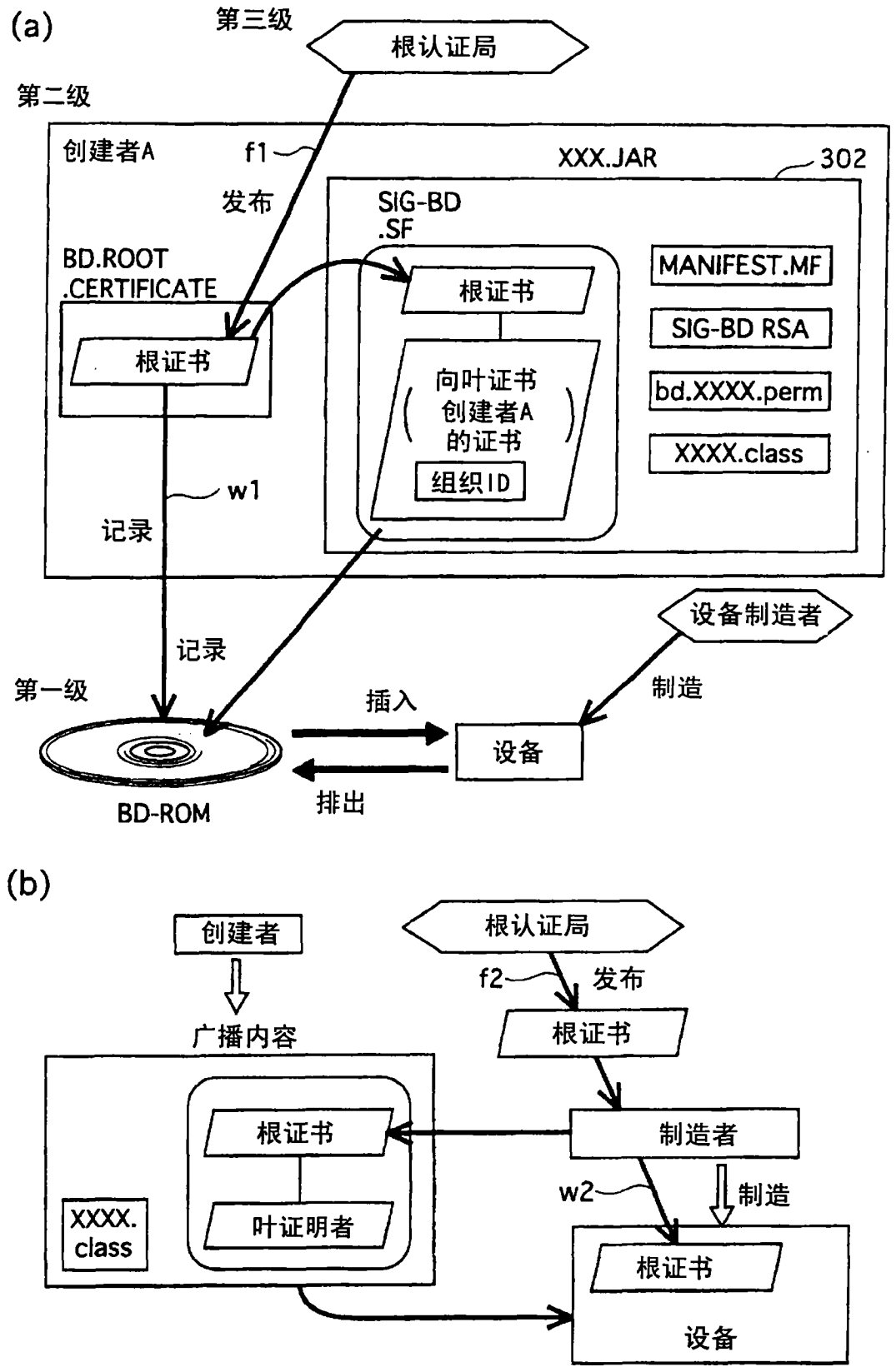

(a)
第三级
根认证局
第二级
创建者A
f1
发布
XXX.JAR
302
SIG-BD
.SF
BD.ROOT
.CERTIFICATE
根证书
根证书
向叶证书
创建者A
的证书
组织ID
MANIFEST.MF
SIG-BD RSA
bd.XXXX.perm
XXXX.class
w1
记录
记录
设备制造者
第一级
插入
制造
设备
排出
BD-ROM
(b)
创建者
广播内容
根认证局
f2
发布
根证书
根证书
叶证明者
制造者
XXXX.
class
w2
制造
根证书
设备

图 3

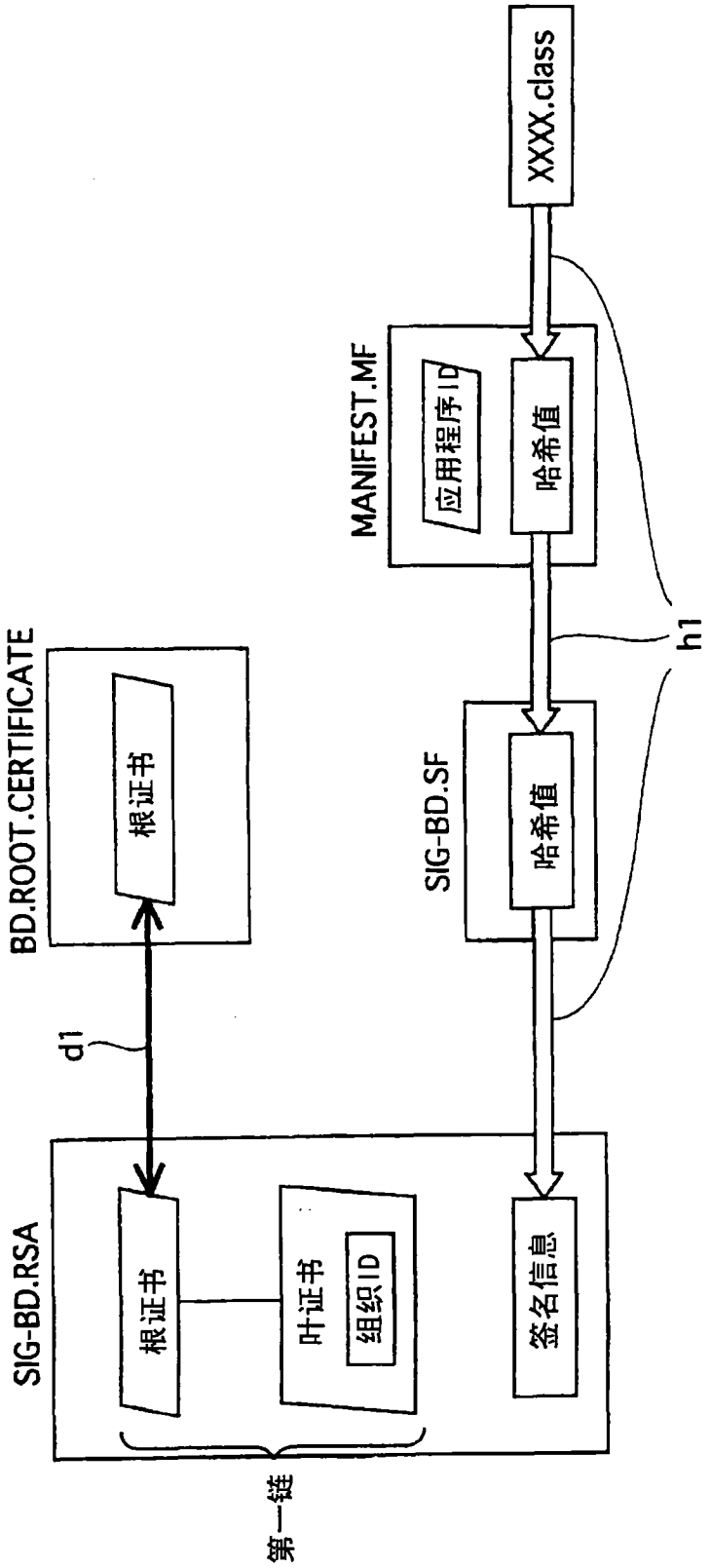

XXXX.class
MANIFEST.MF
应用程序ID
哈希值
h1
BD.ROOT.CERTIFICATE
根证书
SIG-BD.SF
哈希值
d1
SIG-BD.RSA
根证书
叶证书
组织ID
签名信息
第一链

图 4

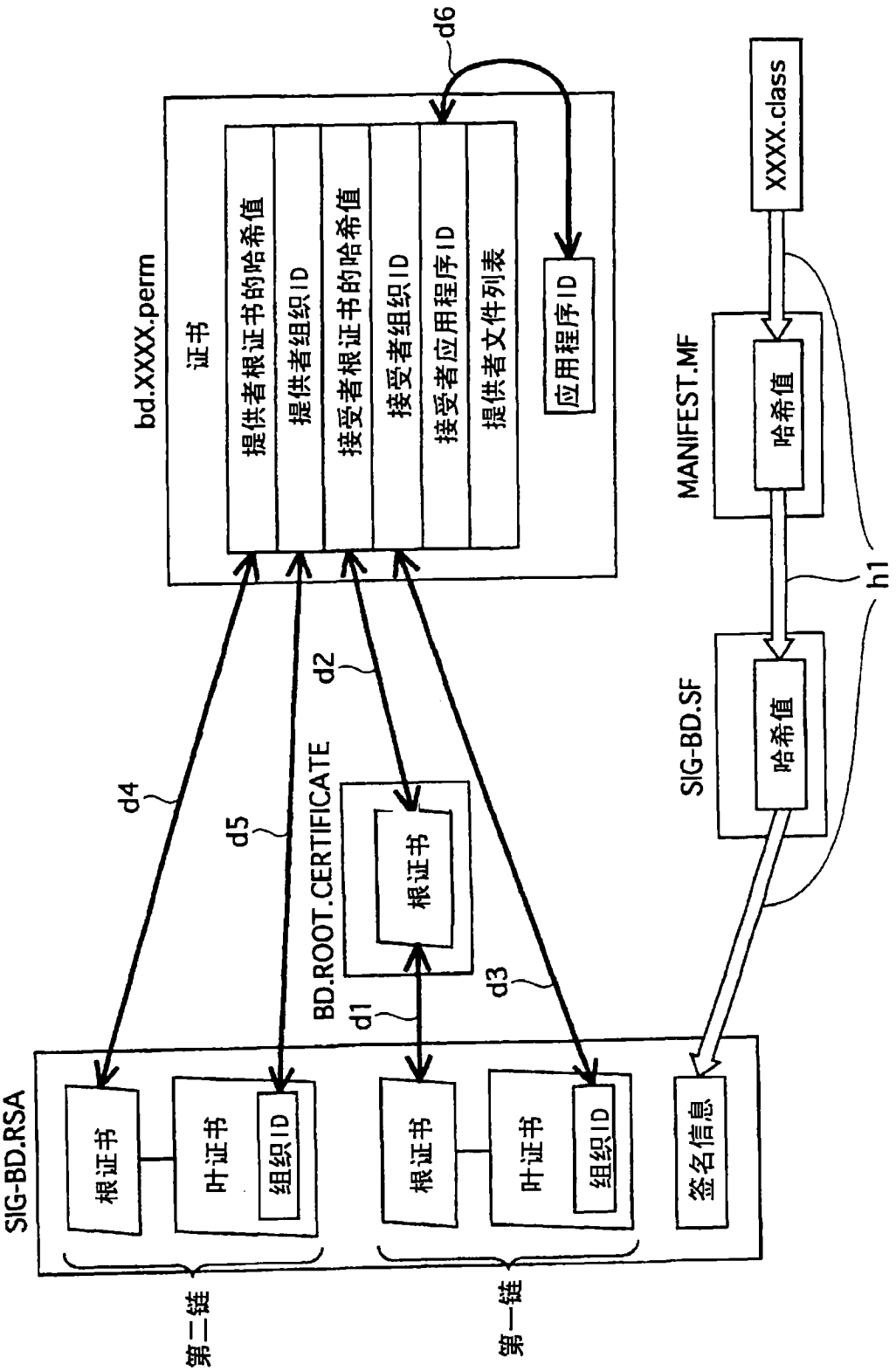

d6
bd.XXXX.perm
证书
提供者根证书的哈希值
提供者组织ID
接受者根证书的哈希值
接受者组织ID
接受者应用程序ID
提供者文件列表
应用程序ID
XXXX.class
MANIFEST.MF
哈希值
h1
SIG-BD.SF
哈希值
d2
d4
d5
BD.ROOT.CERTIFICATE
根证书
d1
d3
SIG-BD.RSA
根证书
叶证书
组织ID
根证书
叶证书
组织ID
签名信息
第二链
第一链

图 5

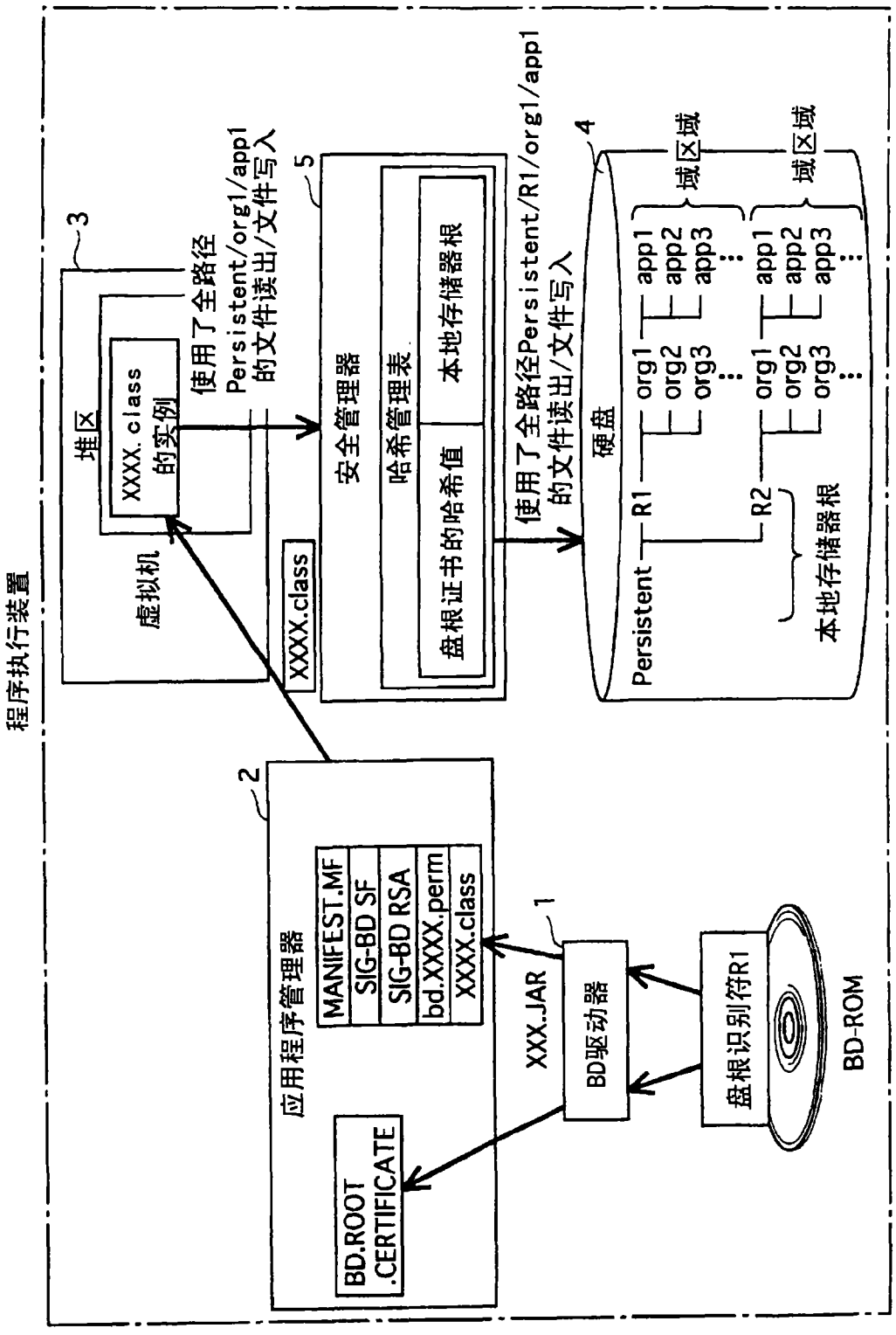

程序执行装置
堆区
XXXX.class
的实例
虚拟机
3
使用了全路径
Persistent/org1/app1
的文件读出/文件写入
XXXX.class
5
安全管理器
哈希管理表
盘根证书的哈希值
本地存储器根
使用了全路径Persistent/R1/org1/app1
的文件读出/文件写入
4
硬盘
Persistent
R1
R2
org1
org2
org3
app1
app2
app3
域区域
本地存储器根
2
应用程序管理器
MANIFEST.MF
SIG-BD SF
SIG-BD RSA
bd.XXXX.perm
XXXX.class
BD.ROOT
.CERTIFICATE
XXX.JAR
1
BD驱动器
盘根识别符R1
BD-ROM

图 6

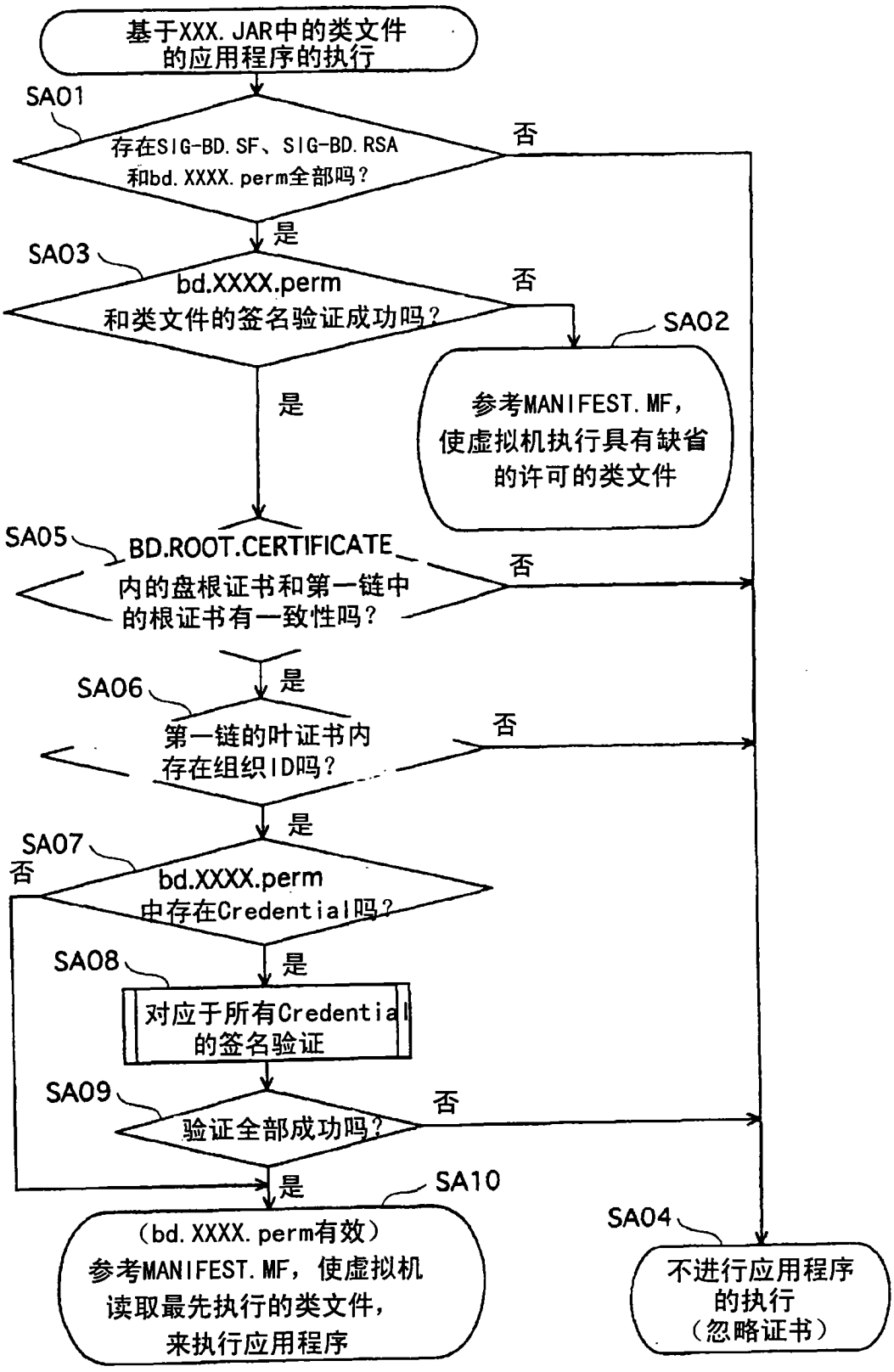

基于XXX.JAR中的类文件的应用程序的执行
SA01
存在SIG-BD.SF、SIG-BD.RSA和bd.XXXX.perm全部吗？
否
是
SA03
bd.XXXX.perm和类文件的签名验证成功吗？
否
SA02
参考MANIFEST.MF，使虚拟机执行具有缺省的许可的类文件
是
SA05
BD.ROOT.CERTIFICATE内的盘根证书和第一链中的根证书有一致性吗？
否
是
SA06
第一链的叶证书内存在组织ID吗？
否
是
SA07
否
bd.XXXX.perm中存在Credential吗？
是
SA08
对应于所有Credential的签名验证
SA09
验证全部成功吗？
否
是
SA10
（bd.XXXX.perm有效）参考MANIFEST.MF，使虚拟机读取最先执行的类文件，来执行应用程序
SA04
不进行应用程序的执行（忽略证书）

图 7

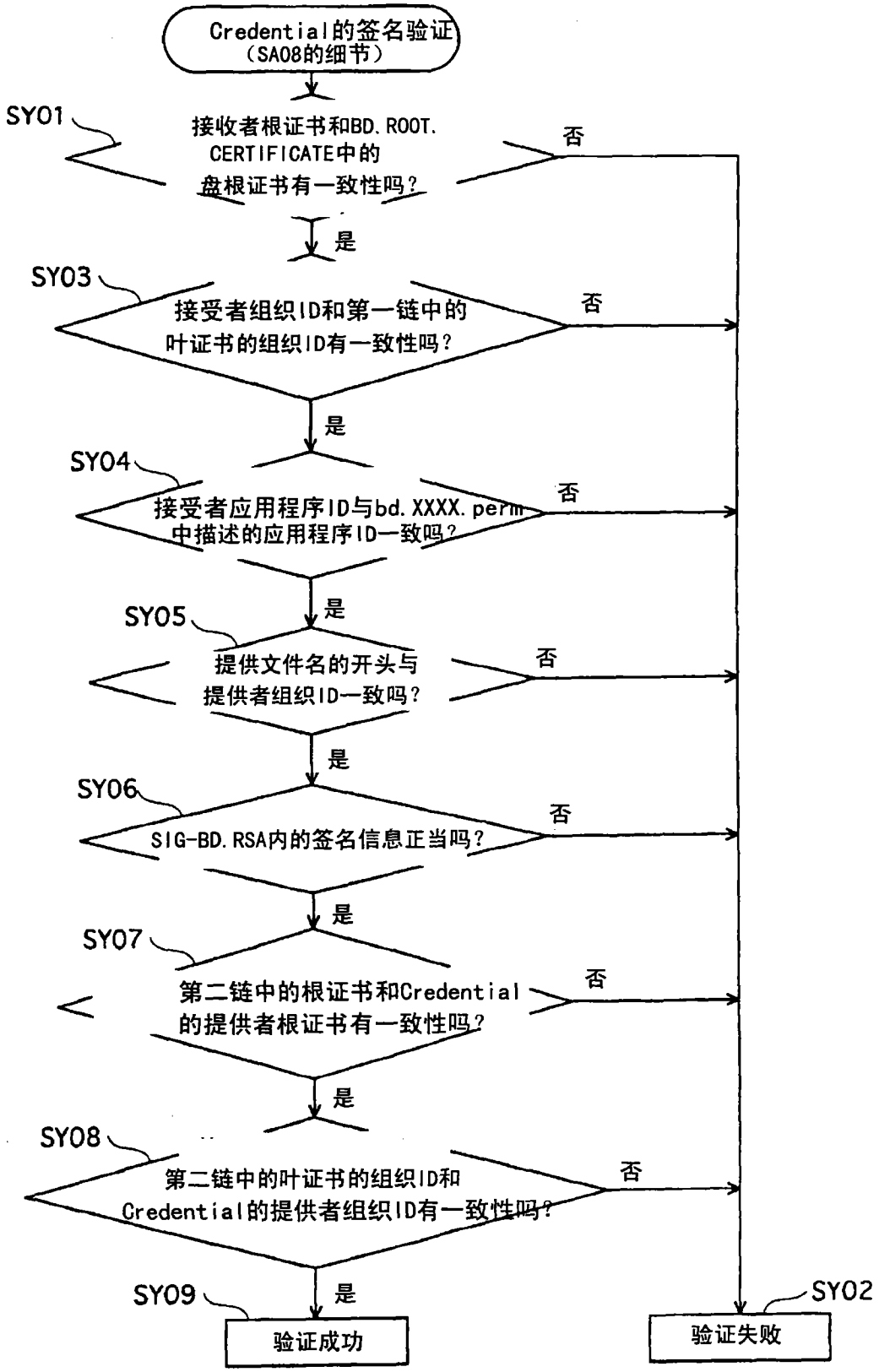

Credential的签名验证
（SA08的细节）
SY01
接收者根证书和BD. ROOT. CERTIFICATE中的盘根证书有一致性吗？
否
是
SY03
接受者组织ID和第一链中的叶证书的组织ID有一致性吗？
否
是
SY04
接受者应用程序ID与bd. XXXX. perm中描述的应用程序ID一致吗？
否
是
SY05
提供文件名的开头与提供者组织ID一致吗？
否
是
SY06
SIG-BD. RSA内的签名信息正当吗？
否
是
SY07
第二链中的根证书和Credential的提供者根证书有一致性吗？
否
是
SY08
第二链中的叶证书的组织ID和Credential的提供者组织ID有一致性吗？
否
是
SY09
验证成功
SY02
验证失败

图 8

应用程序管理器保持的管理信息许可请求文件（bd. XXXX. perm）

盘根证书	(X.509形式の 文件)	302

Jar文件名	“0001.jar”	“0002.jar”
组织ID	“7”	没有
应用程序ID	“8”	“35”
应用程序之间通信	“enable”	“disable”
Credential	Credential列表	没有

Credential列表

图 9

提供者根证书哈希值	96:9E:10:F7:38:53:51:E1 :BD:E8:A8:D1:BA:60:10:94	501
提供者组织ID法	“4”	502
提供者文件列表		506

提供文件名	“4/5/scores.txt”	“4/5/etc/settings.txt”	507
提供者访问方法	“read”	“resad” , “write”	508

图 10

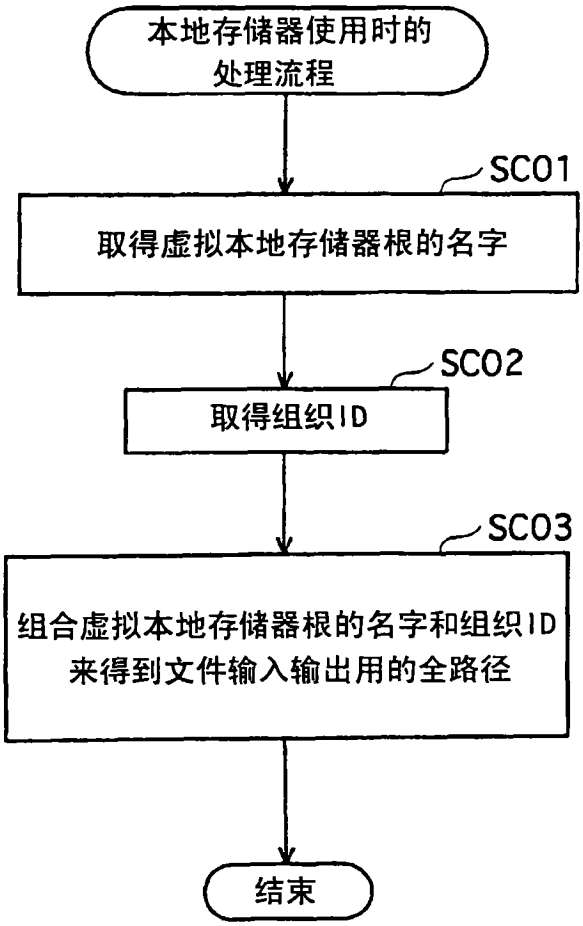

本地存储器使用时的
处理流程
SC01
取得虚拟本地存储器根的名字
SC02
取得组织ID
SC03
组合虚拟本地存储器根的名字和组织ID
来得到文件输入输出用的全路径
结束

图 11

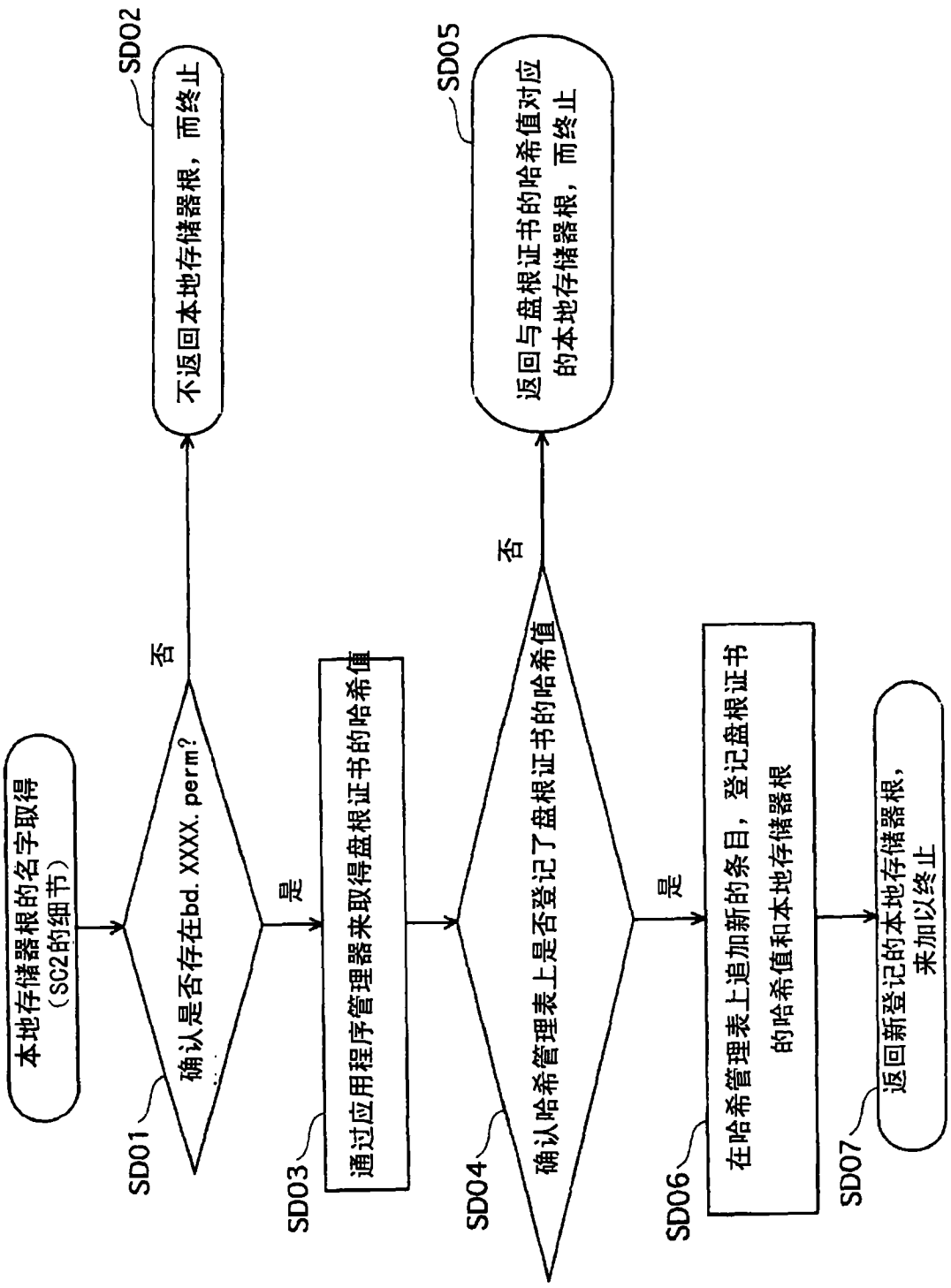

本地存储器根的名字取得
（SC2的细节）
SD01
确认是否存在bd. XXXX. perm?
否
SD02
不返回本地存储器根，而终止
是
SD03
通过应用程序管理器来取得盘根证书的哈希值
SD04
确认哈希管理表上是否登记了盘根证书的哈希值
否
SD05
返回与盘根证书的哈希值对应
的本地存储器根，而终止
是
SD06
在哈希管理表上追加新的条目，登记盘根证书
的哈希值和本地存储器根
SD07
返回新登记的本地存储器根，
来加以终止

图 12

安全管理器保持的哈希管理表

	1301	1302
提供者根证书哈希值	96:9E:10:F7:38:53:51:E1 :BD:E8:A8:D1:BA:60:10:94	42:6C:40:25:2C:4B:04:9D :20:E7:45:97:02:D2:01:08
本地存储器根	/persistent/0001	/persistent/0003

图 13

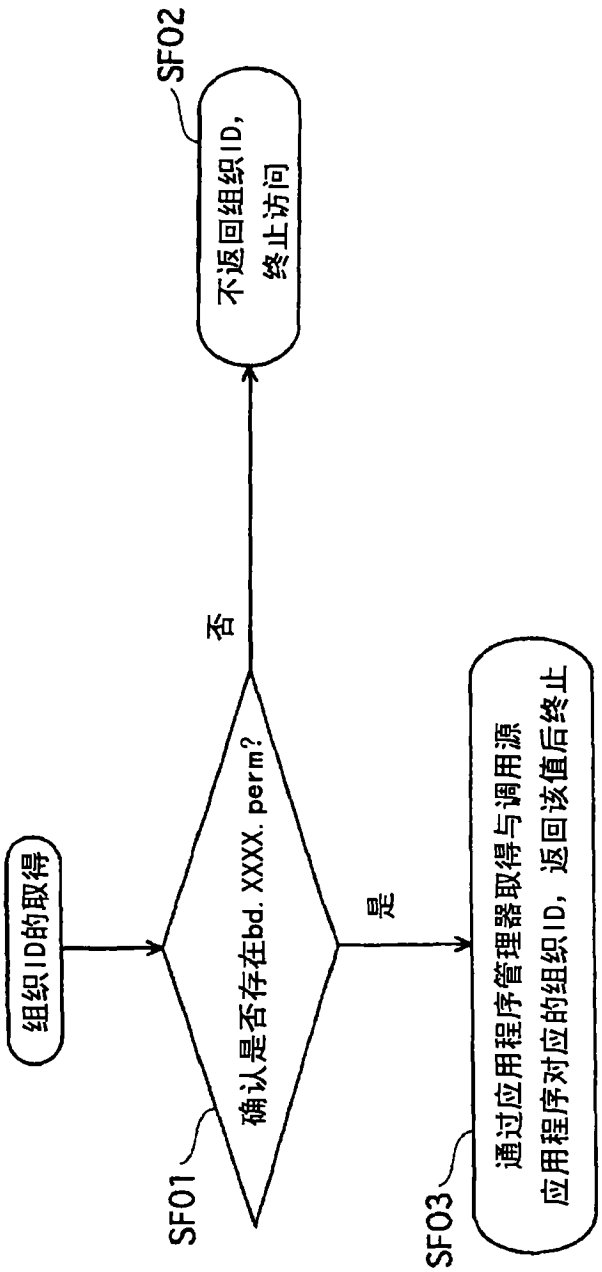


图 14

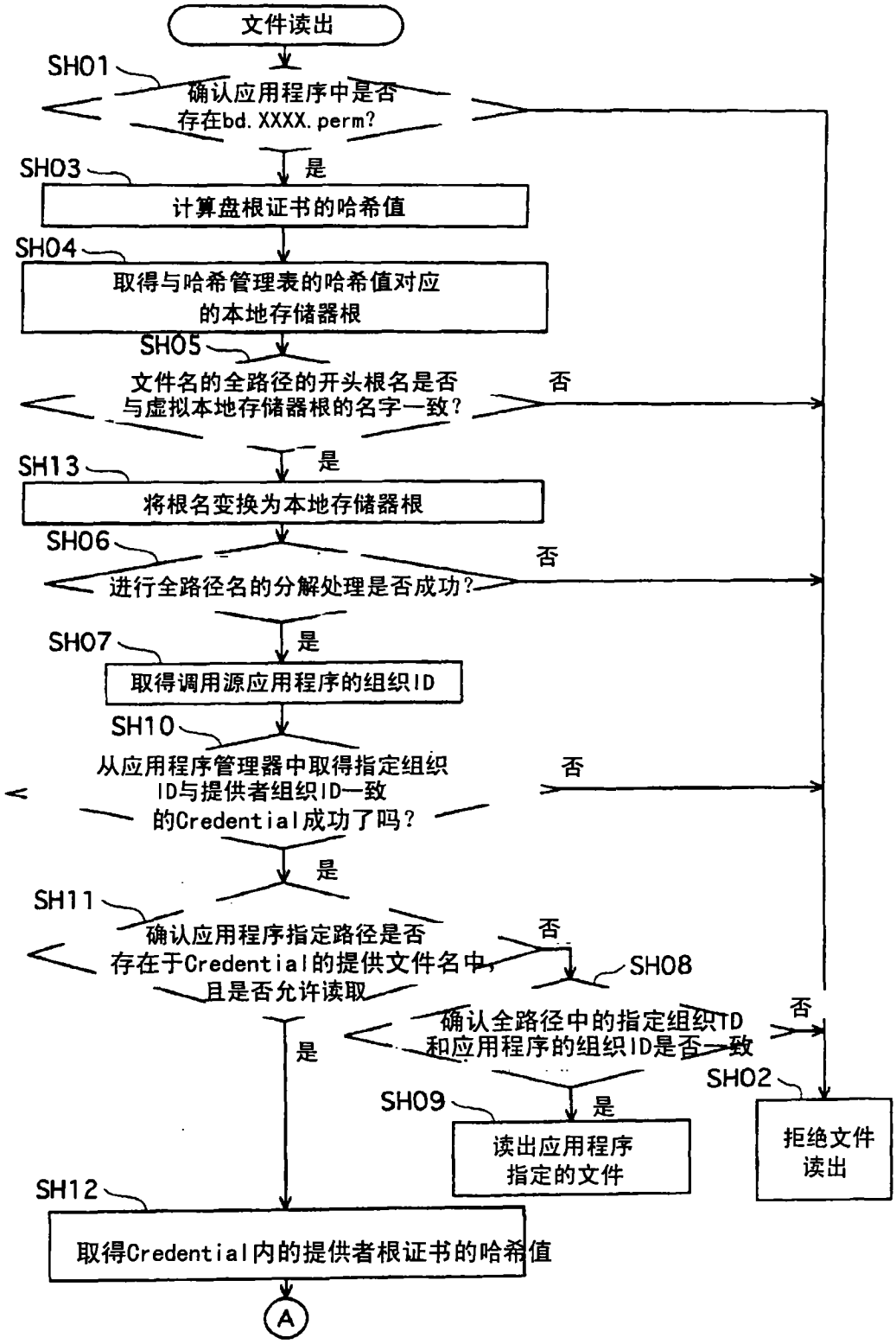

文件读出
SH01
确认应用程序中是否存在bd. XXXX. perm?
是
否
SH03
计算盘根证书的哈希值
SH04
取得与哈希管理表的哈希值对应的本地存储器根
SH05
文件名的全路径的开头根名是否与虚拟本地存储器根的名字一致?
是
否
SH13
将根名变换为本地存储器根
SH06
进行全路径名的分解处理是否成功?
是
否
SH07
取得调用源应用程序的组织ID
SH10
从应用程序管理器中取得指定组织ID与提供者组织ID一致的Credential成功了吗?
是
否
SH11
确认应用程序指定路径是否存在于Credential的提供文件名中,且是否允许读取
是
否
SH08
确认全路径中的指定组织ID和应用程序的组织ID是否一致
是
否
SH09
读出应用程序指定的文件
SH02
拒绝文件读出
SH12
取得Credential内的提供者根证书的哈希值
A

图 15

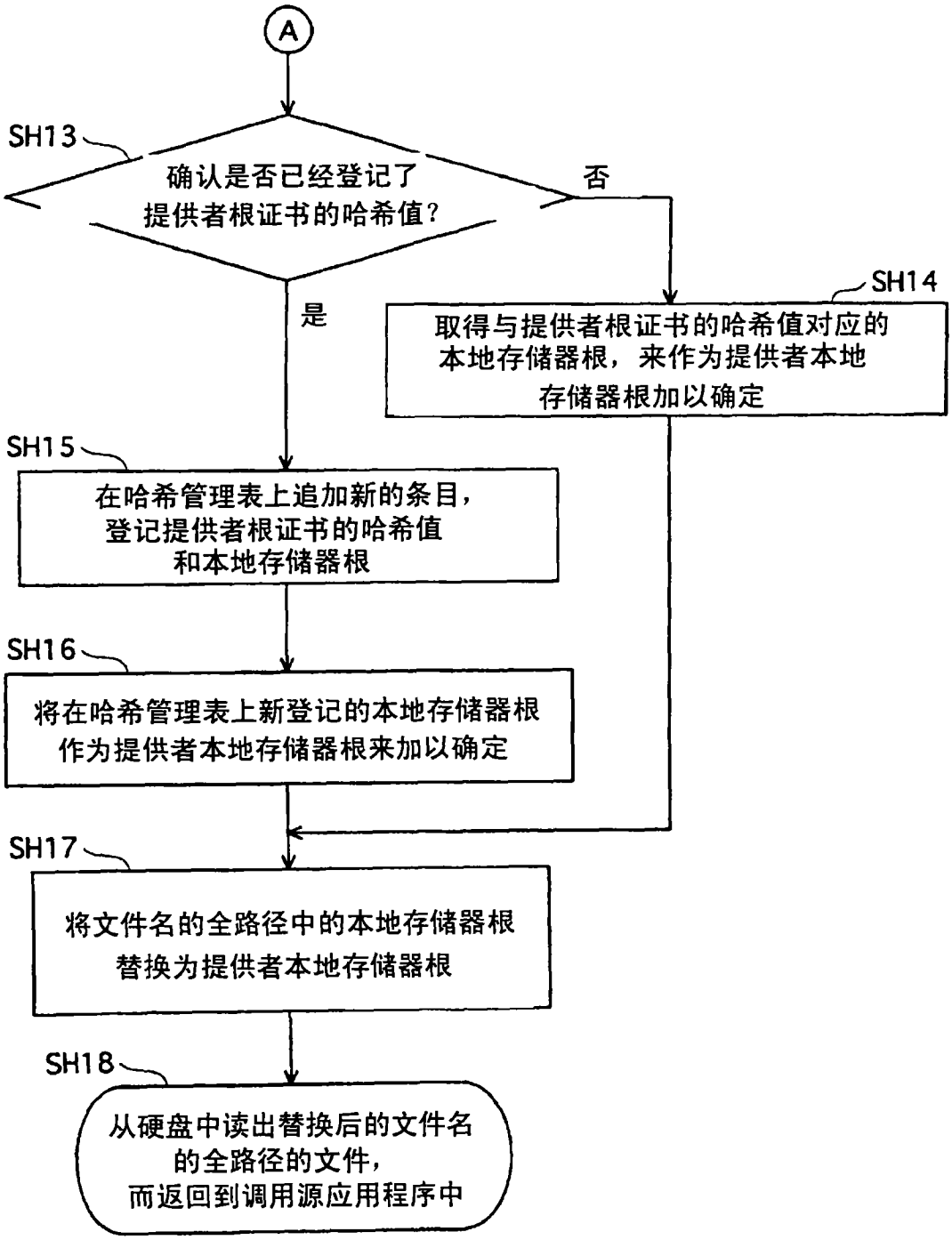

A
SH13
确认是否已经登记了
提供者根证书的哈希值？
否
是
SH14
取得与提供者根证书的哈希值对应的
本地存储器根，来作为提供者本地
存储器根加以确定
SH15
在哈希管理表上追加新的条目，
登记提供者根证书的哈希值
和本地存储器根
SH16
将在哈希管理表上新登记的本地存储器根
作为提供者本地存储器根来加以确定
SH17
将文件名的全路径中的本地存储器根
替换为提供者本地存储器根
SH18
从硬盘中读出替换后的文件名
的全路径的文件，
而返回到调用源应用程序中

图 16

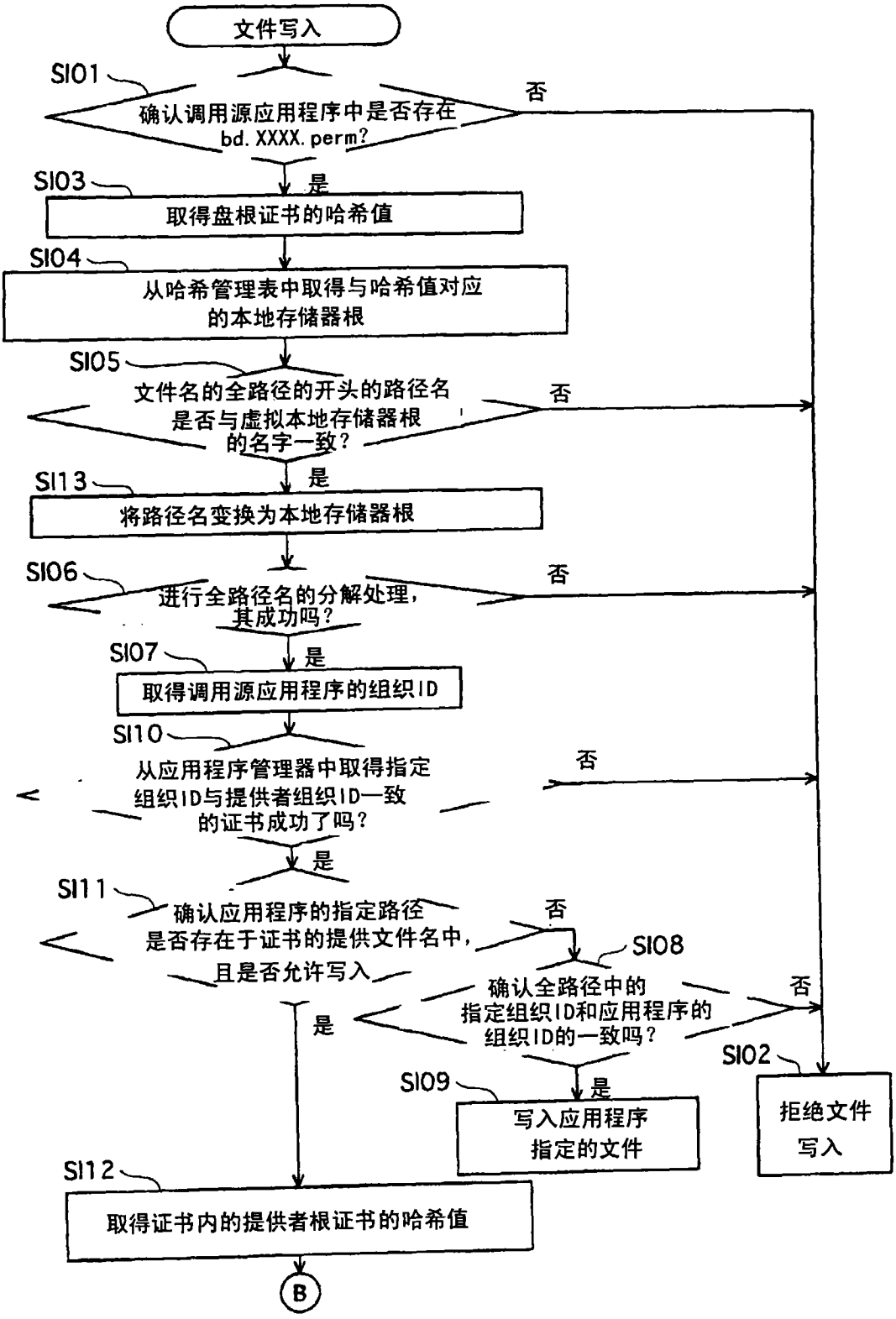

文件写入
S101
确认调用源应用程序中是否存在bd. XXXX. perm?
否
是
S103
取得盘根证书的哈希值
S104
从哈希管理表中取得与哈希值对应的本地存储器根
S105
文件名的全路径的开头的路径名是否与虚拟本地存储器根的名字一致?
否
是
S113
将路径名变换为本地存储器根
S106
进行全路径名的分解处理，其成功吗?
否
是
S107
取得调用源应用程序的组织ID
S110
从应用程序管理器中取得指定组织ID与提供者组织ID一致的证书成功了吗?
否
是
S111
确认应用程序的指定路径是否存在于证书的提供文件名中，且是否允许写入
否
S108
确认全路径中的指定组织ID和应用程序的组织ID的一致吗?
否
是
S102
拒绝文件写入
S109
写入应用程序指定的文件
是
S112
取得证书内的提供者根证书的哈希值
B

图 17

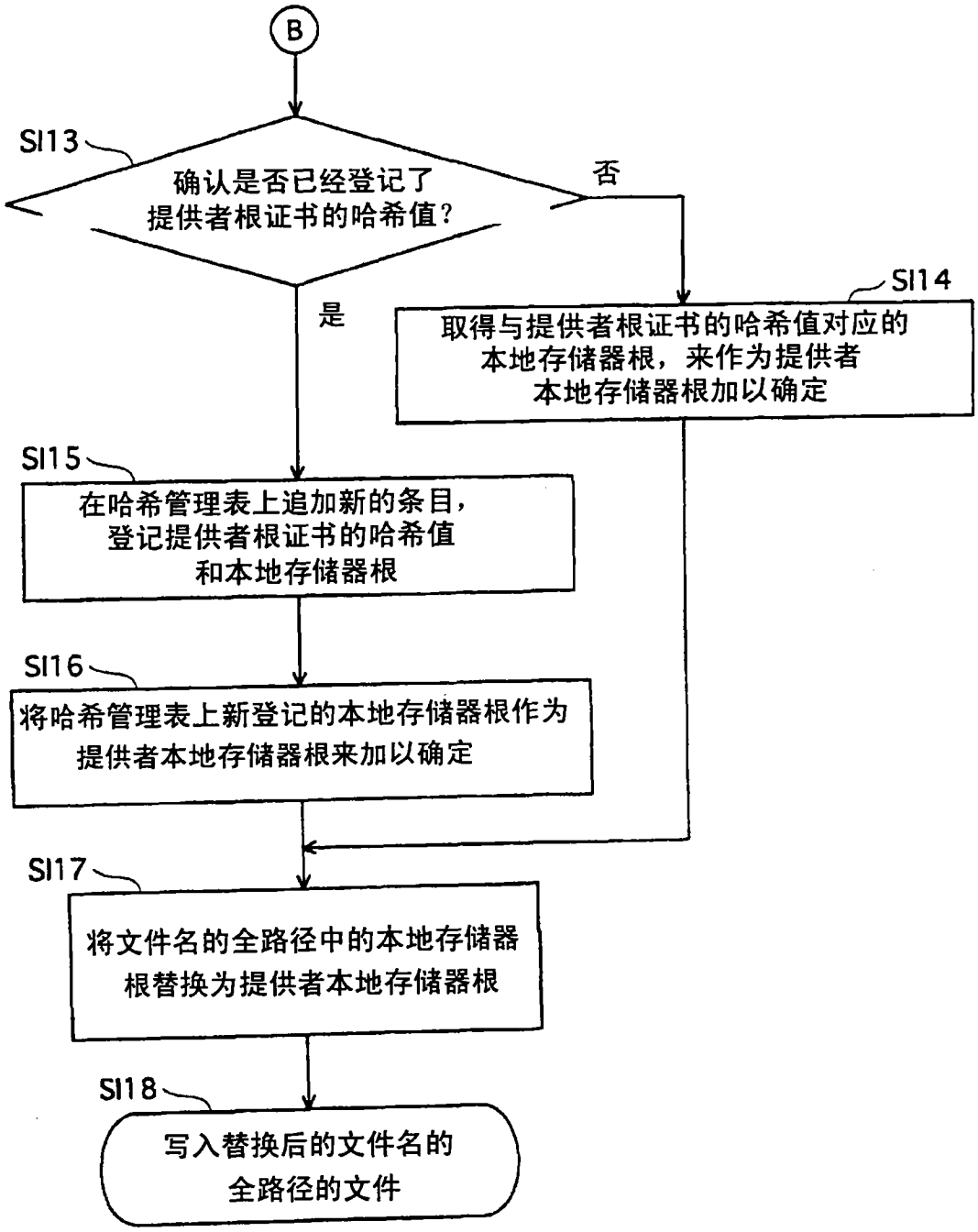

B
SI13
确认是否已经登记了
提供者根证书的哈希值？
否
是
SI14
取得与提供者根证书的哈希值对应的
本地存储器根，来作为提供者
本地存储器根加以确定
SI15
在哈希管理表上追加新的条目，
登记提供者根证书的哈希值
和本地存储器根
SI16
将哈希管理表上新登记的本地存储器根作为
提供者本地存储器根来加以确定
SI17
将文件名的全路径中的本地存储器
根替换为提供者本地存储器根
SI18
写入替换后的文件名的
全路径的文件

图 18