



(12) 发明专利申请

(10) 申请公布号 CN 113169879 A

(43) 申请公布日 2021.07.23

(21) 申请号 201980078378.4

(22) 申请日 2019.11.14

(30) 优先权数据

1819290.6 2018.11.27 GB

1819286.4 2018.11.27 GB

1819284.9 2018.11.27 GB

1819297.1 2018.11.27 GB

1819291.4 2018.11.27 GB

1819293.0 2018.11.27 GB

1819299.7 2018.11.27 GB

(85) PCT国际申请进入国家阶段日

2021.05.27

(86) PCT国际申请的申请数据

PCT/IB2019/059808 2019.11.14

(87) PCT国际申请的公布数据

W02020/109912 EN 2020.06.04

(71) 申请人 区块链控股有限公司

地址 安提瓜和巴布达圣约翰

(72) 发明人 C·S·赖特 J·O·戴维斯

C·C·塔尔坦 O·沃恩

(74) 专利代理机构 隆天知识产权代理有限公司

72003

代理人 石海霞 金鹏

(51) Int.Cl.

H04L 9/32 (2006.01)

权利要求书1页 说明书31页 附图13页

(54) 发明名称

用于在区块链上存储数据的计算机实现的系统和方法

(57) 摘要

公开了一种在诸如比特币区块链等区块链上存储数据的方法。该方法包括生成区块链交易,该区块链交易具有至少一个第一输入和/或至少一个第一输出,以及至少一个第二输入和/或至少一个第二输出,该至少一个第一输入和/或至少一个第一输出包含要存储在区块链上的第一数据(<内容1>),该至少一个第二输入和/或至少一个第二输出包含表示第一数据的至少一个属性的第二数据(<属性1>,<属性2>)。至少一个第二输入和/或至少一个第二输出与该或每个第一输入和/或第一输出分开。

TxID	
输入	输出
<Sig P> <P>	OP_RETURN <Metanet Flag> <Attribute 1> <Attribute 2>
	<Content 1> OP_DROP <H(P')> [CheckSig P']

1. 一种在区块链上存储数据的方法,所述方法包括:

生成区块链交易,所述区块链交易具有至少一个第一输入和/或至少一个第一输出、以及至少一个第二输入和/或至少一个第二输出,所述至少一个第一输入和/或至少一个第一输出包含要存储在区块链上的第一数据,所述至少一个第二输入和/或至少一个第二输出包含表示所述第一数据的至少一个属性的第二数据,其中,至少一个所述第二输入和/或至少一个所述第二输出与所述第一输入和/或第一输出或每个所述第一输入和/或第一输出分开。

2. 根据权利要求1所述的方法,其中,所述第一数据被包含在多个所述第一输入和/或所述第一输出中。

3. 根据权利要求1或2所述的方法,其中,所述第二数据包括与所述第一数据的重组有关的数据。

4. 根据前述权利要求中的任一项所述的方法,其中,至少一个所述第二输入和/或至少一个所述第二输出包括指示数据,所述指示数据指示交易包括所述第一数据。

5. 根据前述权利要求中的任一项所述的方法,其中,所述第二数据表示所述第一数据的至少一个属性,所述至少一个属性是所述第一数据的数据类型、数据加密方案、数据压缩方案、索引信息、许可信息、编码信息、关键词信息或搜索信息中的至少一个。

6. 根据前述权利要求中的任一项所述的方法,其中,所述第一数据被包含在至少一个所述第一输出中。

7. 根据前述权利要求中的任一项所述的方法,其中,至少一个所述第一输出和/或第二输出包括脚本操作码,该脚本操作码用于将输出标记为对于随后用作后续交易的输入是无效的。。

8. 根据前述权利要求中的任一项所述的方法,其中,至少一个所述第一输出是能够通过与应用于所述第一数据的相应数字签名对应的至少一个密钥来兑换的。

9. 根据前述权利要求中的任一项所述的方法,其中,数字签名被应用于所述第一数据。

10. 根据权利要求9所述的方法,其中,至少一个所述数字签名基于具有公钥-私钥对的密码系统,其中,所述私钥基于多个质数,并且对应的公钥基于多个所述质数的乘积。

11. 根据权利要求10所述的方法,其中,至少一个所述数字签名是拉宾签名。

12. 根据前述权利要求中的任一项所述的方法,还包括:将数据压缩应用于所述第一数据和/或所述第二数据。

13. 一种计算机实现的系统,包括:

处理器;以及

包括可执行指令的存储器,所述可执行指令由于被所述处理器执行而使所述系统执行根据权利要求1至12中任一项所述的计算机实现的方法的任何实施方式。

14. 一种非暂时性计算机可读存储介质,其上存储有可执行指令,所述可执行指令由于被计算机系统的处理器执行而使所述计算机系统至少执行根据权利要求1至12中任一项所述的方法的实施方式。

用于在区块链上存储数据的计算机实现的系统和方法

技术领域

[0001] 本发明大体上涉及对跨电子网络,尤其是对等网络(诸如区块链网络等)的改进的数据通信和交换的加密技术。它涉及数据存储、访问、检索和处理,尤其涉及区块链上此类与数据相关的活动。本发明特别适合于但不限于在类似于由网站和网页提供的方式、但是将区块链用作基础机制或平台而不是网络服务器来处理数据时使用。因此,本发明提供了用于数据处理和传送的安全、高效、以密码方式实施的替代基础设施。

背景技术

[0002] 在本文档中,我们使用术语“区块链”来包括所有形式的电子的基于计算机的分布式账本(ledger)。这些包括基于共识的区块链和交易链技术、许可的和未被许可的账本、共享账本及其变型。尽管已经提出并开发了其他区块链实现方式,但是区块链技术最广为人知的应用是比特币账本。尽管为了方便和说明的目的在本文中可能提及比特币,但是应当注意,本发明不限于与比特币区块链一起使用,并且替代的区块链实现和协议落入本发明的范围内。术语“用户”在本文中可以指人或基于处理器的资源。如本文所使用的,“比特币”包括从比特币协议衍生的协议的所有版本和变型。

[0003] 区块链是一种点对点的电子账本,其被实现为基于计算机的去中心化的分布式系统,该系统由区块组成,而区块又由交易组成。每个交易是一种数据结构,该数据结构对区块链系统中参与者之间的数字资产控制权的转移进行编码,并包括至少一个输入和至少一个输出。每个区块包含前一个区块的哈希值,使得区块被链接在一起来创建所有交易的永久、不可更改的记录,这些交易自其开始就已经被写入区块链。交易包含嵌入到其输入和输出中的被称为脚本的小程序,这些小程序指定如何以及由谁可以访问交易的输出。在比特币平台上,这些脚本是使用基于堆栈的脚本语言编写的。

[0004] 为了将交易写入区块链,必须对其进行“验证”。网络节点(矿工)执行工作以确保每笔交易有效,而无效交易则被网络拒绝。安装在节点上的软件客户端通过执行其锁定脚本和解锁脚本来对未花费的交易(unspent transaction, UTXO)执行该验证工作。如果锁定脚本和解锁脚本的执行评估为真(TRUE),则该交易有效,并将该交易写入区块链。因此,为了将交易写入区块链,必须:i)由接收交易的第一节点验证该交易-如果交易经过验证,则该节点将其中继到网络中的其他节点;ii)将该交易添加到由矿工建造的新区块中;以及iii)该交易被挖掘,即,被添加到过去交易的公共账本中。

[0005] 尽管区块链技术因使用加密货币实现方式而被广泛了解,但数字企业家已经开始探索使用比特币所基于的加密安全系统以及可以存储在区块链上的数据这两者以实现新系统。如果区块链可以用于不限于加密货币领域的任务和过程,那将是非常有利的。这样的方案将能够利用区块链的好处(例如,事件的永久性、防篡改记录、分布式处理等),同时在其应用中具有更多用途。

[0006] 一个这样的关注领域是使用区块链来存储、共享、访问和控制用户之间的数据。如今,这经由因特网来实现,其中,服务器托管网站和页面,用户通常通过搜索引擎访问这些

网站和页面,以访问所需的数据。

[0007] 但是,一些观察者已经开始设想使用区块链来解决因特网的某些缺点,例如,由中心化的各方控制大量数据和内容。例如,参见“Life After Google:The Fall of Big Data and the Rise of the Blockchain Economy (谷歌之后的生活:大数据的衰落和区块链经济的崛起)”,George Gilder,Gateway Editions,2018年7月,ISBN-10:9781621575764和 ISBN-13:978-1621575764。

发明内容

[0008] 因此,期望提供一种布置,该布置使得能够有利地利用区块链的分布式、不可变、分布式和永久性的性质在区块链上存储、处理、检索、搜索和/或共享这样的数据。现在已经设计出这种改进的方案。

[0009] 本公开的实施例至少提供了用于实现区块链方案以及用于在其上或从其存储、处理、搜索和/或检索数据的替代的高效和安全的技术。实施例还至少提供了用于在计算节点之间存储、处理、检索、传送、搜索和/或共享数据的替代的区块链实现的技术基础设施。因为本发明使得能够以新的方式使用区块链网络并将其用于提供改进的技术结果,所以本发明提供了一种改进的区块链实现的网络。

[0010] 实施例还提供了用于在技术上不同且改进的计算平台上安全控制对数字资源的访问的方案,该计算平台包括区块链和区块链协议。

[0011] 本发明在所附权利要求书中被限定。

[0012] 根据本发明,可以提供计算机实现的方法。其可以是在区块链上存储数据的方法,该方法包括:

[0013] 生成区块链交易,该区块链交易具有至少一个第一输入和/或至少一个第一输出,以及至少一个第二输入和/或至少一个第二输出,该至少一个第一输入和/或至少一个第一输出包含要存储在区块链上的第一数据,该至少一个第二输入和/或至少一个第二输出包含表示所述第一数据的至少一个属性的第二数据,其中,至少一个所述第二输入和/或至少一个所述第二输出与所述或每个所述第一输入和/或第一输出分开。

[0014] 通过将第一数据与代表第一数据的至少一个属性的第二数据分开存储,这提供了以下优点:使数据能够以独立于用于处理数据的协议的方式存储在区块链上,从而使该过程更普遍适用。另外,将第一数据和第二数据分开提供了以下优点:使第一数据能够更容易地被第三方数字签署。此外,数据属性和内容的明确分开为在区块链中插入数据提供了标准化的程序。这样既可以捕获现有的数据插入方法,又可以为将来的扩展提供支持。此外,一旦将数据插入在区块链上,它就可以具有唯一的标识符和位置。这消除了用户在本地存储数据副本的责任,因为一旦数据被包括在区块链中就不可变。这也意味着用户可以在任何时间点检索数据,而无需任何灾难恢复机制。

[0015] 第一数据可以被包含在多个所述第一输入和/或所述第一输出中。

[0016] 通过将第一数据存储多个第一输入和/或第一输出中,这提供了使得能够最大化交易中的数据存储的密度的优势。

[0017] 所述第二数据可以包括与所述第一数据的重组有关的数据。

[0018] 至少一个所述第二输入和/或至少一个所述第二输出可以包括指示数据,该指示

数据指示交易包括所述第一数据。

[0019] 这提供了以下优势:避免不必要地处理不包括第一数据的区块链交易,从而提高处理效率。

[0020] 第二数据可以表示所述第一数据的至少一个属性,所述至少一个属性是所述第一数据的数据类型、数据加密方案、数据压缩方案、索引信息、许可信息、编码信息、关键词信息或搜索信息中的至少一个。

[0021] 第一数据可以被包含在至少一个所述第一输出中。

[0022] 这提供了以下优势:使得能够检查交易的有效性,作为区块链挖掘过程的一部分。通过基础区块链的现有的矿工验证过程确保数据完整性。这允许区块链用作受信任的全局服务器,在全局服务器处,存储的数据的完整性通过矿工保证,从而其可以由其他各方进行存储和分发,而无需信任。

[0023] 至少一个所述第一输出和/或第二输出可以包括脚本操作码,其用于将输出标记为对于随后用作后续交易的输入是无效的。

[0024] 这提供了以下优势:避免将输出作为未花费交易输出 (UTXO) 数据的一部分进行处理,从而提高处理效率。

[0025] 至少一个所述第一输出和/或第二输出可以是可花费的。

[0026] 这提供了以下优势:使可执行程序能够被包括在第一数据中,否则其可能被阻止在不可花费的输出进行操作,例如作为OP_RETURN运算符的结果。OP_RETURN运算符还具有终止可执行程序的附加优点。

[0027] 至少一个所述第一输出可以是借助于与应用于所述第一数据的相应数字签名对应的至少一个密钥可兑换的。

[0028] 这提供了以下优势:利用验证交易的数字签名的过程,其作为加密货币挖掘过程的一部分执行来验证交易是有效的,以进一步验证第一数据和/或第二数据的有效性。

[0029] 数字签名可以应用于所述第一数据。

[0030] 这提供了以下优势:使得第一数据能够更容易地与第二数据隔离,并且能够独立于第二数据由第三方进行数字签署,从而使该过程更通用,并且使得能够更快地处理数据。这也使该方法能够更独立于协议。

[0031] 至少一个所述数字签名可以基于具有公钥-私钥对的密码系统,其中,该私钥基于多个质数,并且对应的公钥基于多个所述质数的乘积。

[0032] 这提供了以下优势:提供了签名方案,该签名方案对于验证而言在计算上不昂贵,从而提高了处理效率。

[0033] 至少一个所述数字签名可以是拉宾签名。

[0034] 这提供了以下优势:分发者可以使用拉宾签名从原始数据源提供认证。这允许内容本身由原始作者进行签署以证明其真实性。

[0035] 该方法还可以包括将数据压缩应用于所述第一数据和/或所述第二数据。

[0036] 本发明还提供一种系统,该系统包括:

[0037] 处理器;以及

[0038] 包括可执行指令的存储器,该可执行指令由于被处理器执行而使得系统执行本文所述的计算机实现的方法的任何实施例。

[0039] 本发明还提供了其上存储有可执行指令的非瞬时性计算机可读存储介质,该可执行指令由于被计算机系统的处理器执行而使得计算机系统至少执行本文所述的计算机实现的方法的实施例。

附图说明

[0040] 本发明的这些和其他方面将从本文描述的实施例变得显而易见并参考这些实施例而阐明。现在将仅通过示例的方式并参考附图来描述本发明的实施例,附图中:

[0041] 图1示出了体现本发明的区块链交易,在该区块链交易中,数据被存储在多个输出中;

[0042] 图2示出了体现本发明的区块链交易,在该区块链交易中,数据被存储在输入中。

[0043] 图3示出了体现本发明的一系列区块链交易,在这些区块链交易中,数据被存储在多个区块链交易的输出上;

[0044] 图4示出了体现本发明的区块链交易,该区块链交易传送加密货币支付以允许借助于原子交换来访问数据;

[0045] 图5示出了体现本发明的区块链交易,用于兑换图4的交易的支付;

[0046] 图6示出了在体现本发明的区块链交易中由参与者所保存的秘密值,该区块链交易发布令牌以允许借助于原子交换来访问数据;

[0047] 图7和图8示出了体现本发明的区块链交易,用于发布令牌以允许借助于原子交换来访问数据;

[0048] 图9和图10示出了体现本发明的区块链交易,用于兑换借助于图7和图8的交易发布的令牌;

[0049] 图11和图12示出了用于访问由图9和图10的交易交换的秘密的区块链交易;

[0050] 图13提供了根据本发明实施例的元网图结构的图示。

[0051] 图14示出了根据本发明实施例的用于包括MURL搜索路径的域“bobsblog”的元网图树的图示。

[0052] 图15示出了根据本发明的一个示例的浏览器钱包以及如何在应用的不同组件之间分割其核心功能的说明性实施例的示意图。

[0053] 图16提供了示出如何在本发明实施例的基础设施内执行搜索内容的图。

[0054] 图17示出了根据本发明实施例的本地全副本对等方和全局全副本对等方之间的说明性交互。

[0055] 图18示出了元网树(或图),以供参考下面描述的说明性用例使用。

[0056] 图19示出了流程图,该流程图说明了下面提供的说明性用例所体现的过程。

[0057] 图20是示出其中可以实现各种实施例的计算环境的示意图。

具体实施方式

[0058] 术语“比特币”在本文中仅出于方便目的而使用,并且旨在包括所有加密货币/区块链协议,包括但不限于:衍生自比特币协议的所有变型以及其他区块链的任何替代协议。在本文档的其余部分中,确定本发明实施例的操作的协议将被称为“元网协议”。

[0059] 根据本发明的实施例,术语“内容”和“数据”在本文中可以互换使用,以指代存储

在区块链交易中的数据。

[0060] 概述

[0061] 如上所述,已经认识到需要一种改进的和/或替代的基础设施,以在计算节点之间并通过计算节点存储、写入、访问和查看数据。使用区块链技术固有的优势(例如,不可变记录、以密码方式实施的控制和访问、内置支付机制、公开检查账本的能力、分布式架构等)将是有利的。但是,从许多技术角度来看,构建“区块链实现的互联网”具有挑战性。

[0062] 这些挑战可能包括但不限于:如何定位网络中数据的特定部分;如何保护和控制对数据的访问,以使只有授权方才能获得访问权限;如何以点对点方式将数据从一方传送到另一方;如何安排数据,使得其可以在逻辑上关联但仍存储在网络内的不同位置中,以及随后如何将其从不同位置进行组合以提供总体和增强的结果;如何以分层方式提供和/或存储数据;如何允许具有不同计算平台的用户和各方访问所需的数据;如何跨(潜在地全球)计算网络存储、提供和共享数据,而不依赖或不需要大型存储服务器和集中式数据控制器,以及如何提高网络上此类数据相关活动的效率。

[0063] 在本文中,“共享”可以包括向节点或用户提供、发送、传达、传输数据的部分或提供对数据的部分的访问。术语“处理”可以解释为表示与交易或其相关联数据有关的任何活动,包括生成、传输、验证、访问、搜索、共享提交到区块链网络和/或标识。

[0064] 本发明以这样的方式提供了这种改进的方案:该方式在某些方面类似于因特网,但是使用与现有技术中已知的完全不同的硬件组件和软件组件的平台,以完全不同的方式实现其结果。根据本发明的实施例,存储互联网/网络数据并将其提供给最终用户的服务器被驻留在区块链网络上的区块链交易所替换。为了实现这一点,必须设计出若干创新。这些将在以下各部分中进行描述。

[0065] 将数据插入到区块链“元网”中参照图1,示出了体现本发明的区块链交易,在该区块链交易中,要存储在区块链上的第一数据存储在一个或多个第一输出中,并且表示第一数据的属性的第二数据存储在一个或多个第二输出中。第一数据的一个或多个第一部分<内容1>存储在交易的可花费输出中。表示第一数据的相应的属性的数据<属性1>和<属性2>以及指示正在根据元网协议存储数据的标志存储在交易的第二不可花费输出中。术语“不可花费”用于指示交易的至少一个第一和/或第二输出可以包括脚本操作码(OP RETURN),该脚本操作码用于将输出标记为对于随后用作后续交易的输入是无效的。

[0066] 将数据的内容和属性部分分别地存储在交易的单独输出中是有利的。

[0067] 图2示出了体现本发明的区块链交易,在该区块链交易中,要存储在区块链上的第一数据<内容1>存储在交易的输入中。元网标志以及属性数据<属性1>和<属性2>以类似于图1所示的布置的方式存储在交易的不可花费输出中。

[0068] 数据插入

[0069] 数据插入方法

[0070] 希望能够将以下数据插入到区块链中

[0071] a) 元网标志

[0072] b) 属性

[0073] c) 内容

[0074] 内容是要存储在区块链上的数据,元网标志是4字节的前缀,其充当与元网协议有

关的任何数据的标识符,而属性包含有关内容的索引、许可和编码信息。这可以包括但不限于数据类型、加密和/或压缩方案。这样的属性通常也称为元数据。为了避免与交易元数据混淆,将避免在本文档中使用该术语。

[0075] 以下技术可以用于将该数据嵌入到比特币脚本中:

[0076] 1.OP_RETURN-在这种方法中,所有数据(属性和内容)在可证明的不可花费交易输出的锁定脚本中都放置在OP_RETURN之后。

[0077] 使用该运算符的输出脚本示例如下:

[0078] UTX00:OP_RETURN<Metanet Flag><attributes><content>

[0079] 2.带有OP_DROP的OP_RETURN-在这种情况下,OP_RETURN包含属性,而内容在可花费交易脚本(锁定或解锁)中存储在OP_DROP之前。内容可以在交易输入和输出中被分割成多个数据包。但是,将数据插入到交易输出中是有利的,因为仅输出脚本可以在比特币协议中被签署。如果将数据插入到交易输入中,则OP_MOD可以代替矿工验证用作对数据的校验和,以确保其有效性。例如,可以执行32位OP_MOD运算并检查其是否等于预先计算的值。

[0080] 在这种情况下,属性可以包含有关如何重组内容数据包的信息。另外,将重组的数据包H(内容1+内容2)的哈希提供为属性使得能够验证推荐的重组方案已被使用。

[0081] 图1中示出了实现第二数据插入方法的交易。为简单起见,该交易仅包括插入其输出中的内容,该输出由其单个输入签署。使用图2所示的该方法来使用OP_DROP语句也可以将内容插入到另外的输入中。

[0082] 如果内容很大,则将其在多个交易上进行分割可能是有利的。这样的布置在图3中示出。图3示出了体现本发明的成对的区块链交易,在该成对的区块链交易中,要存储在区块链上的第一数据<内容>被分割成两个组块<内容组块1>和<内容组块2>,这两个组块随后可以重组为<内容>=<内容组块1>||<内容组块2>,其中,运算符“||”将两个组块的内容数据串连。该串连运算符可以被任何所需的按位或类似的分段的二进制运算符代替。然后这两个组块<内容组块1>和<内容组块2>存储在单独的区块链交易的相应的可花费输出中,而与内容数据的属性有关的数据存储在区块链交易的相应的不可花费输出中。再次,属性可以包含有关重组方案的信息。例如,内容可以是原始数据、可执行程序或HTML网页。另外,内容1可以包括指向内容2在区块链上的位置的指针,其以与网页内的嵌入式HTML链接相同的方式起作用。

[0083] 应当注意,两个交易将采用同一公钥P(和ECDSA签名)作为输入,使得尽管<内容组块1>和<内容组块2>存储在分别具有TxID1和TxID2的不同交易中,但是它们可以通过同一公钥P相关。

[0084] 利用矿工验证的作用

[0085] 此处,矿工执行的交易验证过程用于在存储此数据时获得优势。这是因为交易输出中的所有数据将由公钥P的所有者在至少一个交易输入中签署(如果存在SIGHASH|ALL标志),并且该签名将在所有矿工执行的交易验证过程中被检查。

[0086] 这样确保

[0087] • 数据完整性-若数据损毁,则CHECKSIG操作将失败。

[0088] • 数据真实性-P的所有者已可证明地见证并签署数据。

[0089] 这对于在多个交易上被分割的内容尤其有利,因为P的输入签名提供了数据的分

割分量之间的可证明链接,如上文参考图3中所示的布置所描述的。

[0090] 拉宾(Rabin)签名

[0091] 确保数据真实性的另一个方式是使用Rabin签名,其可用于签署数据本身而非整个消息。这可以是有利的,因为签署者无需签署出现数据的每个单独的交易,且签名可在多个交易中重复使用。

[0092] Rabin签名可以在脚本中容易地验证。通过将Rabin签名验证插入在OP_DROP命令之前,可以将这些并入在上文的情况(2)中,即

[0093] $\langle \text{content1} \rangle \langle \text{Rabin Sig}(\text{content1}) \rangle \text{FUNC_CHECKRABSIG OP_DROP} \langle \text{H}(\text{P}_1) \rangle$
[CheckSigP₁]

[0094] 应注意,这无法在上文的情况(1)中进行,因为不论如何,包含OP_RETURN的脚本皆失败,因此无法实现验证。

[0095] 使用Rabin签名的具体示例

[0096] 介绍

[0097] 数字签名是比特币协议的基本部分。其确保区块链上记录的任何比特币交易均已被发送比特币的合法持有者授权。在标准比特币P2PKH交易中,使用椭圆曲线数字签名算法(ECDSA)签署交易消息。然而,ECDSA签名通常应用于整个交易。

[0098] 存在比特币区块链的一些用例,其中,来自网络外部的参与者可能想要为任意数据类型提供签名,然后网络参与者可使用该数据类型。通过使用Rabin数字签名,可对任何数据段进行签署-即使其来源于比特币区块链外部,然后被放置在一或多个交易中。

[0099] 现在将示出可如何通过利用Rabin密码系统的代数结构直接以比特币脚本签署及验证数据。

[0100] Rabin数字签名

[0101] Rabin数字签名算法

[0102] 背景数学

[0103] 定义-整数mod p

[0104] 整数模p被定义为如下集合

$$[0105] \quad \mathbb{Z}_p := \{1, 2, \dots, p-1\}$$

[0106] 费马小定理

[0107] 令p为质数。然后对于任何整数a,适用以下条件

$$[0108] \quad a^{p-1} \equiv 1 \pmod{p}$$

[0109] 欧拉准则

[0110] 令p为质数。当且仅当满足下式时,r是二次余数mod p

$$[0111] \quad r^{\frac{p-1}{2}} \equiv 1 \pmod{p}$$

[0112] 模平方根 (p=3mod 4)

[0113] 令p为质数,使得p=3mod 4。那么对于任何满足欧拉准则的整数r,如果a为整数,则使得

$$[0114] \quad a^2 \equiv r \pmod{p}$$

[0115] 然后,a存在形式为下式的解

[0116]
$$a \equiv \pm r^{\frac{p+1}{4}} \pmod{p}$$

[0117] 中国剩余定理

[0118] 给定成对的互质正整数 $n_1, n_2, \dots, n_k$ 和任意整数 $a_1, a_2, \dots, a_k$, 联立同余 (simultaneous congruences) 的系统

[0119] $x \equiv a_1 \pmod{n_1}$

[0120] $x \equiv a_2 \pmod{n_2}$

[0121] $\vdots$

[0122] $x \equiv a_k \pmod{n_k}$

[0123] 具有唯一的解模数 $N = n_1 n_2 \dots n_k$ 。作为中国剩余定理的特例, 可以示出:

[0124] 当且仅当

[0125] $x \equiv r \pmod{n_1 \cdot n_2}$ 时,

[0126] $x \equiv r \pmod{n_1}$ 且 $x \equiv r \pmod{n_2}$

[0127] Rabin数字签名算法

[0128] Rabin数字签名算法可以如下描述:

[0129] 对于任何消息 m , 令 H 为具有 k 个输出位的抗冲突哈希算法。

[0130] 为了生成密钥, 选择质数 p 和 q , 每个质数的位长近似为 $k/2$, 使得 $p \equiv 3 \pmod{4}$, $q \equiv 3 \pmod{4}$ 并计算乘积 $n = p \cdot q$ 。私钥为 (p, q) , 公钥为 $n = p \cdot q$ 。

[0131] 为了签署消息 m , 签署者选择填充 U , 使得 $H(m || U)$ 满足

[0132]
$$H(m || U)^{\frac{p-1}{2}} \equiv 1 \pmod{p}$$

[0133]
$$H(m || U)^{\frac{q-1}{2}} \equiv 1 \pmod{q}$$

[0134] 签名 S 使用以下公式计算

[0135]
$$S \equiv \left[\left(p^{q-2} \cdot H(m || U)^{\frac{q+1}{4}} \pmod{q} \right) \cdot p + \left(q^{p-2} \cdot H(m || U)^{\frac{p+1}{4}} \pmod{p} \right) \cdot q \right] \pmod{n}。$$

[0136] 消息 m 的签名是对 (S, U) 。验证可以通过对于给定的 m, U 和 S , 检查下式来简单地进行

[0137] $H(m || U) \equiv S^2 \pmod{n}$ (等式1)。

[0138] 当且仅当在范围 $0, \dots, n-1$ 内存在整数 λ , 使得

[0139] $H(m || U) + \lambda \cdot n = S^2$ (等式2) 时, 这为真。

[0140] 因子 λ 可以安全地包括在签名中, 以提供组合 (S, λ, U) 。

[0141] Rabin签名方案的有利特征如下:

[0142] a) 签名生成在计算上是昂贵的, 而签名的验证在计算上是容易的。

[0143] b) 签名的安全性仅依赖于整数分解的难度。因此, Rabin签名在本质上是不可伪造的 (与RSA不同)。

[0144] c) 哈希函数值 $H(m || U)$ 必须具有与公钥 n 相似的大小。

[0145] 脚本中的验证简单明了, 因为它只需要对给定签名进行平方, 执行模归约, 然后检

查结果是否等于 $H(m||U)$ 。

[0146] Rabin签名证明

[0147] 令 p, q 为互质数, 且 $n=p \cdot q$ 。通过中国剩余定理, 可以示出:

[0148] 当且仅当

[0149] $S^2 \equiv H(m||U) \pmod{p}$

[0150] $S^2 \equiv H(m||U) \pmod{q}$ 时,

[0151] 则

[0152] $S^2 \equiv H(m||U) \pmod{n}$

[0153] 使用以下等式

$$S \equiv \left(\left(p^{q-2} H(m||U)^{\frac{q+1}{4}} \pmod{q} \right) \cdot p + \left(q^{p-2} H(m||U)^{\frac{p+1}{4}} \pmod{p} \right) \cdot q \right) \pmod{q}$$

$$\equiv \left(\left(p^{q-2} H(m||U)^{\frac{q+1}{4}} \pmod{q} \right) p \right) \pmod{q}$$

[0154]

$$\equiv p^{q-2} H(m||U)^{\frac{q+1}{4}} \cdot p \pmod{q}$$

$$\equiv (p^{q-1} \pmod{q}) H(m||U)^{\frac{q+1}{4}} \pmod{q}$$

$$\equiv H(m||U)^{\frac{q+1}{4}} \pmod{q}$$

[0155] 可以示出:

[0156] $S^2 \equiv H(m||U) \pmod{q}$

[0157] 所以

[0158] $S^2 \equiv H(m||U)^{\frac{q+1}{2}} \equiv H(m||U)^{\frac{q-1}{2}} \cdot H(m||U) \equiv H(m||U) \pmod{q}$

[0159] 其中, 已经假设 $H(m||U)$ 满足欧拉准则。通过类似的计算, 还可以示出:

[0160] $S^2 \equiv H(m||U) \pmod{p}$ 。

[0161] 比特币中的Rabin签名

[0162] 脚本中的签名验证

[0163] 需要少量算术和堆栈操纵操作码来验证Rabin签名。考虑以下形式的兑换脚本

[0164] OP_DUP OP_HASH160< $H_{160}(n)$ >OP_EQUALVERIFY OP_MUL OP_SWAP OP_2OP_ROLL
OP_CAT FUNC_HASH3072 OP_ADD OP_SWAP OP_DUP OP_MUL OP_EQUAL

[0165] 其中, n 是签名者的公钥。当且仅当具备如下的输入时, 这将评估为真

[0166] $\langle S \rangle \langle U \rangle \langle m \rangle \langle \lambda \rangle \langle n \rangle$

[0167] 其中, m 是任意消息, (S, λ, U) 是有效的Rabin签名。可替代地, 如果使用上面的等式1检查了Rabin签名, 则兑换脚本由下式给出:

[0168] OP_DUP OP_HASH160 $\langle H_{160}(n) \rangle$ OP_DUP OP_TOALTSTACK OP_SWAP $\langle \text{roll index} \rangle$ OP_ROLL OP_CAT FUNC_HASH3072 OP_SWAP OP_MOD OP_SWAP OP_DUP OP_MUL OP_FROMALTSTACK OP_MOD OP_EQUAL

[0169] 在这种情况下, 当且仅当具备如下的输入时, 这将评估为真

[0170] $S \rangle \langle U \rangle \langle m \rangle \langle n \rangle$

[0171] 在这两个兑换脚本中, 均使用了3072位哈希投影函数“FUNC_HASH3072”。对于给定的消息/填充串连, 使用如下脚本生成FUNC_HASH3072哈希投影:

[0172] OP_SHA256 {OP_20P_SPLIT OP_SWAP OP_SHA256 OP_SWAP} (x11)

[0173] OP_SHA256 OP_SWAP OP_SHA256 {OP_CAT} (x11)

[0174] 数据压缩

[0175] 因特网数据由JavaScript和常见的文件类型 (例如, 文本文件 (SML、HTML等)、视频文件 (MPEG、M-JPEG等)、图像文件 (GIF、JPEG等) 和音频文件 (AU、WAV等) 组成, 例如, 如在以下链接中更详细地描述的: https://www.doc.ic.ac.uk/~nd/surprise_97/journal/voll/mmp/#text。使用上述数据插入技术, 这些不同的数据类型也可以嵌入在区块链上。

[0176] 在将较大文件大小嵌入于区块链上之前, 可使用若干现有的编码方案中的一个对其进行压缩。诸如游程长度及霍夫曼 (Huffman) 编码等无损数据压缩算法可用于若干应用, 包括ZIP文件、可执行程序、文本文档及源代码。

[0177] 取决于具体的输入数据, 存在许多不同的算法。苹果无损及自适应变换声编码可用于压缩音频文件, PNG及TIFF用于压缩图形文件, 而电影文件可使用许多无损视频编译码器中的一个进行压缩。可使用属性内的标志来指示数据内容的任何压缩。例如, 属性中用于LZW无损编码方案的标志将为 $\langle \text{LZW} \rangle$ 。

[0178] 加密和支付的解密

[0179] 数据加密

[0180] 内容的所有者可选择在将内容嵌入在区块链上之前保护内容。这确保了在未获取到必要许可的情况下无法查看内容。

[0181] 存在许多公认的用于加密数据 (明文或其他数据类型) 的技术。这些技术可归类为非对称加密或对称加密。

[0182] 椭圆曲线密码术 (ECC) 是非对称的, 因为其依赖于公钥-私钥对。其为最安全的密码系统之一, 并且通常用于诸如比特币等加密货币中。对于ECC密码术, Koblitz算法可用于加密数据。

[0183] 在对称方案中, 单个密钥用于既加密又解密数据。高级加密标准 (AES) 算法被视为由这样的秘密作为种子产生的最安全的对称算法之一, 例如如以下文献中更详细地描述的: C.Paar及J.Pelzl, “Understanding Cryptography (理解密码术)” 中的第4章, Springer-Verlag Berlin Heidelberg, 第2版, 2010年, 第87至118页。

[0184] 当加密存储于区块链上的数据时, 使用与基础区块链相同的密码系统存在优势。在比特币中, 这为非对称密码术中的ECC密钥对的secp256k1惯例, 及对称密码术中的SHA-

256哈希函数。这些优势为：

- [0185] -加密的安全级别与其上存储有数据的基础系统相同。
- [0186] -存储加密数据所需的软件架构将具有较小的代码库。
- [0187] -钱包中的密钥管理可用于交易和加密/解密两者。
- [0188] -由于相同的密钥可用于加密和以加密货币的支付，因此效率更高，因此所需的密钥更少。这也减少了存储空间。
- [0189] -交换/购买解密数据的能力可能需要更少的通信信道。
- [0190] -由于用于加密和交易的密钥是相同的数据结构，因此提高了安全性，因此减轻了对于特定密钥类型的针对性攻击。
- [0191] -可以使用基础加密货币购买密钥。

[0192] 为了说明的目的，描述可以如何使用Koblitz算法以使用ECC来加密数据。

[0193] Koblitz算法

[0194] 给定ECC密钥对 $P_1 = S_1 \cdot G$ ，Koblitz算法允许任何人使用公钥 P_1 来加密消息，使得仅已知对应私钥 S_1 的人可解密消息。

[0195] 假设期望使用Koblitz方法加密明文消息‘hello world’。这是逐字符进行的。如下加密及解密第一字符‘h’。

[0196] 1. 字符‘h’映射到secp256k1曲线上的点。这可以通过使用ASCII惯例将明文字符映射到8位数字来实现。然后通过将基点 G 乘以该数字来计算曲线上的点。在本示例中，‘h’映射到ASCII中的104，椭圆曲线点由 $P_m = 104 \cdot G$ 给出。

[0197] 2. 然后使用公钥 P_1 对点 P_m 进行加密。这通过选择随机临时密钥 k_0 并计算点对 $C_m = \{k_0 \cdot G, Q\}$ （其中， $Q := P_m + k_0 \cdot P_1$ ）来实现，然后可以广播该点对。

[0198] 3. 私钥 S_1 的所有者可以通过计算 $P_m = Q - S_1 \cdot k_0 \cdot G$ 来解密原始点。然后，他们可以通过试错法或借助于查找表来恢复原始ASCII数字，以确立哪个数字 x 对应于 $P_m = x \cdot G$ 。

[0199] 使用区块链来购买许可

[0200] 将数据存储在区块链上具有支付机制构建在系统中的明显优势。支付可用于购买

[0201] -解密数据以便查看/使用

[0202] -在特定地址处插入数据的许可

[0203] 在这两种情况下，买方均使用加密货币（例如，比特币）来购买授予其进行某事的许可的秘密。该秘密可以是哈希原像或私钥。

[0204] 进行这种购买的高效且安全的方式是使用原子交换。这将安全通信信道保持为最小，并确保向卖方支付且向买方揭露秘密，或确保没有任何事件发生。

[0205] 除了以加密货币支付外，使用访问令牌购买许可也很方便。这是买方拥有的其可使用以便进行购买的秘密值（通常为哈希原像）。买方可事先大量购买这样的访问令牌，然后在其实际上希望使用许可时进行激活。

[0206] 现在将参考图4和图5描述如何执行原子交换。

[0207] 使用哈希谜题或私钥谜题的原子交换

[0208] 假设爱丽丝是秘密的所有者。该秘密可以是已知哈希摘要的哈希原像，或已知公钥的私钥。假设鲍勃希望使用比特币来从爱丽丝购买该秘密。描述称为原子交换的机制，该机制使此交易能够发生。在爱丽丝获得支付的比特币且向鲍勃揭露秘密、或者没有任何事

件发生的意义上,这是原子的。

[0209] 该方法如下:

[0210] 爱丽丝拥有公钥/私钥对 $P_A = S_A \cdot G$ 的私钥 S_A ,鲍勃拥有公钥/私钥对 $P_B = S_B \cdot G$ 的私钥 S_B 。

[0211] 爱丽丝拥有作为已知哈希摘要 $H(X)$ 的原像 X 或已知公钥 $P_1 = S_1 \cdot G$ 的私钥 S_1 的秘密。

[0212] 他们同意爱丽丝以比特币价格将该秘密卖给鲍勃。

[0213] 在此之前,鲍勃必须设置交易以在区块外将临时密钥 k_0 发送给爱丽丝,使得爱丽丝可以计算数字签名的分量 r_0 。

[0214] 现参考图4,

[0215] 1. 鲍勃将通过以下兑换脚本 R (示意性地编写)锁定的比特币转移到爱丽丝:

[0216] 对于哈希原像:

[0217] $R = [\text{Hash PuzzleH}(X)] [\text{CheckSigP}_A]$

[0218] 这迫使原像 X 在兑换脚本的输入中暴露。

[0219] 对于私钥:

[0220] $R = [\text{Private Key PuzzleP}_1, r_0] [\text{CheckSigP}_A]$

[0221] 这迫使私钥 S_1 能够从兑换脚本的输入中计算出来。在这种情况下,鲍勃和爱丽丝必须就用于构造 r_0 的临时密钥 k_0 达成一致,其中, $(r_0, R_y) = k_0 \cdot G$ 。

[0222] 2. 由于爱丽丝知道她的秘密(X 或 S_1),因此她可以借助于图5所示的交易花费其在比特币区块链上的资金。这使鲍勃可以确定她的秘密。

[0223] 作为可选的安全特征,爱丽丝和鲍勃可使用他们的公钥 P_A, P_B 来确立仅他们两方已知的共享秘密 S 。这可以以国际专利公开号W0 2017/145016中概述的方式实现。在这种情况下, S 可以被添加至哈希谜题中的原像 X ,以便不在区块链上公开揭露 X 。类似地,在私钥谜题中, S 可用作临时密钥 k_0 以确保仅爱丽丝或鲍勃能够计算私钥。

[0224] 如果爱丽丝决定不花费其资金,则可在程序中引入时间锁定的退款以防止鲍勃的资金被爱丽丝锁定。

[0225] 使用令牌购买

[0226] 假设存在如上文所描述的相同情况,但并非在使用时以加密货币支付爱丽丝的秘密,而是鲍勃希望兑换提前购买的访问令牌,以交换秘密。

[0227] 爱丽丝和鲍勃必须遵循的程序类似于先前章节中描述的情况,但是使用了类似的原子交换的序列。该过程存在两个阶段:令牌发布和令牌兑换。

[0228] 阶段1:令牌发布

[0229] 令牌发布阶段实际上是鲍勃单次购买令牌。例如,考虑如下场景:爱丽丝具有10个不同的秘密 $X_1, X_2, \dots, X_{10}$,并且鲍勃希望单次购买各自授予其对相应的秘密的访问权限的10个令牌 $T_1, T_2, \dots, T_{10}$ 。

[0230] 首先,鲍勃从仅其已知的秘密种子值生成10个令牌的集合。这些令牌是通过种子的顺序的哈希创建以形成哈希链,其中,每个令牌被计算为:

[0231] $T_i = H^{10-i}(Y)$ 对于 $i \in \{1, 2, \dots, 10\}$ 。

[0232] 爱丽丝和鲍勃现在各自具有10个秘密值,这些秘密值可在哈希谜题中揭露以用于

例如兑换令牌。然而,为了发布这些令牌,他们也必须分别生成秘密初始化值 I_{Alice} 和 I_{Bob} 。这些值如下给出:

[0233] $I_{\text{Alice}} = k, \quad k \in \mathbb{Z}_{256},$

[0234] $I_{\text{Bob}} = H^{10}(Y)。$

[0235] 应注意的是,爱丽丝的初始化值(initialiser)简单地为无特定含义的随机整数,但是鲍勃的初始化值应为其第一令牌的哈希 $T_1 = H^9(Y)$ 。以此方式将令牌链扩展至初始化值允许令牌发布也限定稍后待用于相继兑换的令牌。图6中示出了由每一个参与者保存的全部秘密值。

[0236] 现在爱丽丝和鲍勃可以同意以10个加密货币单位的价格购买10个令牌。这些令牌的购买可以通过多种方式进行,此处使用原子交换进行说明。原子交换通过爱丽丝及鲍勃分别广播图7和图8中所示的交易而开始,在这两个交易中输出均需要两个哈希谜题的解和有效签名。

[0237] 一旦区块链中出现两个交易,爱丽丝及鲍勃就可以共享其共享初始化值 I_{Alice} 和 I_{Bob} ,并完成令牌发布的原子交换。

[0238] 由于该原子交换,爱丽丝接收购买10个令牌的支付并且两个初始化值秘密被揭露。应注意的是,此处仅鲍勃的秘密 $I_{\text{Bob}} = H^{10}(Y)$ 是有意义的,因为其将限定待求解的第一哈希谜题[哈希谜题(T_1)] ,其解为初始化值 $H^{10}(Y)$ 的原像 $H^9(Y)$ 。

[0239] 阶段2:令牌兑换

[0240] 在未来的某一时刻,鲍勃想要兑换其第一令牌 $T_1 = H^9(Y)$ 并接收其第一秘密 X_1 ,但是前面已经提到其已通过购买有效令牌支付了该秘密。兑换令牌的过程将采用另一个原子交换的形式,其中,锁定哈希谜题的解为令牌 T_1 及对应的秘密 X_1 。

[0241] 为了兑换其令牌,鲍勃应广播图9中所示的交易,该交易的输出用两个哈希谜题锁定。当爱丽丝看到此交易时,她广播如图10中所示的她自己的类似交易,该交易的输出用相同的两个哈希谜题锁定。两个参与者现在可以交换其秘密 T_1 和 X_1 ,并解锁这些交易的输出。两方现在可通过提供也曝露两秘密的正确的解锁脚本来兑换标称费用 x 。图11及图12中示出具有这些解锁脚本的交易。

[0242] 用于兑换令牌的该原子交换的完成向鲍勃揭露爱丽丝的第一秘密 X_1 ,向爱丽丝揭露鲍勃的第一令牌 T_1 ,并且鉴于金额 x 适当大以鼓励两方花费锁定的输出,具有加密货币资金的净零交换。至关重要地,这也确立了鲍勃可使用的下一个令牌必须是哈希谜题[哈希谜题 $H(T_2)$]的解 T_2 ,其中,刚刚已向爱丽丝揭露了目标哈希 $H(T_2) = T_1$ 。可以递归地重复该过程,直到鲍勃已使用其最后的令牌 $T_{10} = Y$ 为止。

[0243] 命名和寻址

[0244] 节点和边缘结构

[0245] 上面已经解释了可以如何通过交易内提供数据而将数据插入到区块链中。现在呈现用于以逻辑方式结构化这些交易的协议,该逻辑方式允许对节点寻址、许可及内容版本控制。该分布式对等元网的结构类似于现有的因特网。

[0246] 应注意的是,这为不修改基础区块链的协议或共识规则的“tier-2”协议。

[0247] 此处描述的结构的目标为:

[0248] (i) 将不同交易中的相关内容相关联,以实现数据的搜索、标识和访问

[0249] (ii) 允许使用人类可读的关键字搜索来标识内容,以提高搜索速度、准确性和效率

[0250] (iii) 在区块链中构建和模拟类似于服务器的结构

[0251] 我们的方法是将与元网相关联的数据结构化化为定向图。该图的节点及边缘对应于:

[0252] 节点-与元网协议相关联的交易。节点存储内容。(术语“内容”和“数据”在本文档内可互换使用)。

[0253] 节点通过包括由就在<Metanet Flag>之前的OP_RETURN而创建。每个节点被分配有公钥 P_{node} 。公钥与交易ID的组合唯一地指定节点的索引 $ID_{node}: H(P_{node} || TxID_{node})$ 。

[0254] 所使用的哈希函数应符合本发明将与例如比特币的SHA-256或RIPEMD-160一起使用的基础区块链协议。

[0255] 边缘-子节点与父节点的关联。

[0256] 边缘在签名 $SigP_{parent}$ 出现在元网交易的输入中时被创建,因此仅父代可以给予创建边缘的许可。所有节点可以具有至多一个父代,并且父节点可以具有任意数量的子代。在图论的语言中,每个节点的入度(indegree)至多为1,每个节点的出度(outdegree)是任意的。

[0257] 应注意,边缘为元网协议的态样(aspect)且其自身并不是与基础区块链相关联的交易。

[0258] 由具有以下形式的交易给出有效元网节点(具有父代):

$TxID_{node}$	
输入	输出
[0259] $\langle Sig P_{parent} \rangle$ $\langle P_{parent} \rangle$	OP_RETURN $\langle Metanet Flag \rangle$ $\langle P_{node} \rangle$ $\langle TxID_{parent} \rangle$

[0260] 该交易包含指定以下的节点及其父代的索引所需的所有信息:

[0261] $ID_{node} = H(P_{node} || TxID_{node})$,

[0262] $ID_{parent} = H(P_{parent} || TxID_{parent})$ 。

[0263] 此外,由于需要父节点的签名,因此仅父代可创建到子代的边缘。如果 $\langle TxID_{parent} \rangle$ 字段并不存在或其并不指向有效的元网交易,则节点是孤儿。该节点没有可以到达它的更高级别的节点。

[0264] 附加属性可以被添加到每个节点。这些属性可包括标志、名称及关键字。稍后在本文档中讨论这些属性。

[0265] 如所示出的,节点(交易)的索引可以划分为

[0266] a) 公钥 P_{node} ,我们将其解释为节点的地址

[0267] b) 交易ID $TxID_{node}$,我们将其解释为节点的版本

[0268] 这种结构化产生了两个有利的特征:

[0269] 1. 版本控制-如果存在具有相同公钥的两个节点,则我们将具有交易ID的节点解释为该节点的最新版本,该交易ID具有最大工作量证明。如果节点在不同的区块中,则这可

以通过区块高度来进行检查。对于相同区块中的交易,这通过拓朴交易排序规则(TTOR)来确定。

[0270] 2. 许可-仅当公钥 P_{node} 的所有者在创建子节点时签署交易输入时,才可创建节点的子代。因此, P_{node} 不仅表示节点的地址,而且还表示创建子节点的许可。这有意地类似于标准比特币交易—公钥不仅是地址,而且是与该地址相关联的许可。

[0271] 应注意,由于父节点的签名出现在UXT0解锁脚本中,因此在网络接受交易时通过标准矿工验证过程进行验证。这意味着创建子节点的许可由比特币网络自身验证。

[0272] 值得注意的是,标准因特网协议(IP)地址仅在某一时间点时在网络内是唯一的。另一方面,元网中的节点的索引在所有时间都是唯一的,并且不存在单独的网的概念,这允许数据永久性地锚定至单个对象 ID_{node} 。

[0273] 节点和边缘结构允许将元网可视化如图形,如图13所示。

[0274] 元网中的域、命名和定位内容

[0275] 元网图的层次结构允许出现丰富的类似域的结构。我们将孤儿节点解译为顶级域(TLD),将孤儿节点的子代解译为子域,将孙代解译为子子域等,并且将无子节点解译为端点(end-point)。参见图13。

[0276] 域名被解译为 ID_{node} 。元网中的每个顶级域可被视为树,其中,根为孤儿节点,叶为无子节点。元网自身为形成图的树的全局集合。

[0277] 元网协议不规定任何节点包含内容数据,但叶(无子)节点表示数据树上的定向路径的端部,因此将通常用于存储内容数据。然而,内容可存储在树中的任何节点处。节点中作为属性包括的协议特定的标志可用于指定数据树中的节点的作用(磁盘空间、文件夹、文件或许可改变)。

[0278] 前面已经提到因特网使用域名系统(DNS)来将人类可读的名称关联到因特网协议(IP)地址。DNS在某种意义上是去中心化的,尽管在实践中其由少量的关键参与者(例如,政府和大公司)控制。取决于你的DNS提供商,同一名称可将你带至不同的地址。在将人类可读的短名称映像到计算机生成的数字时,该问题是固有的。

[0279] 我们假定存在将人类可读的顶级域名映射到根节点的去中心化索引 ID_{root} 的等效分布式系统。换句话说,存在将人类可读的名称映像到元网根节点索引的1-1函数 κ ,例如:

[0280] $\kappa('bobsblog') = ID_{\text{bobsblog}} (=H(P_{\text{bobsblog}} || TxID_{\text{bobsblog}}))$

[0281] 向左侧的输入是人类可读的词,而右侧的输出为哈希摘要,其将通常为256位数据结构。应注意, P_{bobsblog} 和 $TxID_{\text{bobsblog}}$ 通常也不是人类可读的。在标准IP协议中,这将为从www.bobsblog.com到网络内对应的域的IP地址的映射。

[0282] 映射 κ 应被解译为用于确保在复制DNS发布的域名的人类可读性时元网与因特网的向后兼容性的措施,但提供元网的结构命名及寻址方案并不明确地取决于该映射。

[0283] 映射函数的可能的现有形式包括由星际文件系统(IPFS)或OpenNIC服务(<https://www.openic.org>)采用的DNSLink系统。该映射可作为DNS的一部分存储在现有的TXT记录中。这类似于IPFS中的DNSLink,请参见<https://docs.ipfs.io/guides/concepts/dnslink/>。然而,大体而言,这些牺牲了一些去中心化的元素以便提供作为1-1的映射,请参见<https://hackernoon.com/ten-terrible-attempts-to-make-the-inter-planetary-file-system-human-friendly-e4e95df0c6fa>

[0284] 虚荣地址

[0285] 用作元网节点的地址的公钥不是人类可读的对象。这可能使人类用户的搜索、引用及输入活动易于出错且很慢。然而,可以创建人类可识别的公钥地址—虚荣地址 P_{vanity} —其包括可由用户直接解译的明文前缀。虚荣地址在现有技术中是已知的。

[0286] 创建这种地址的难度取决于所需前缀的字符长度。这意味着人类可识别的虚荣地址可用作节点地址,该节点地址仅依赖于所有者的创建工作量而非中心发布。对于给定前缀,由于后缀中剩余的字符,存在许多不同的虚荣地址,因此许多节点地址可共享公共前缀,同时仍保持唯一性。

[0287] 具有所需前缀的虚荣地址的示例为:

[0288] P_{bobsblog} :bobsblogHtKNngkdXEeobR76b53LETtpyT

[0289] 前缀:bobsblog

[0290] 后缀:HtKNngkdXEeobR76b53LETtpyT

[0291] 上面的虚名地址可用于感测检查从名称‘bobsblog’到节点索引 ID_{bobsblog} 的映射并辅助元网节点通过地址的可搜索性。应注意,前缀在此处并不是唯一的,但整个地址自身是唯一的实体。

[0292] 选择的地址 P_{vanity} 与TxID一起形成 ID_{node} 的组合也是有益的,因为这意味着不存在域名的中心发布者(TxIDs由去中心化的工作量证明生成),并且名称是可从区块链本身恢复的。有利地,不会再有存在于因特网DNS内的故障点。

[0293] 由于元网域已经提供了许可系统(公钥),因此不需要发布凭证以证明所有权。已经例如在域名币(<https://namecoin.org/>)中探索将区块链用于该目的。然而,根据本发明,不需要将单独的区块链用于该功能,因为在一个区块链内能实现所有事情。

[0294] 相比于现有技术,这显著地减小了本发明所需要的资源(硬件、处理资源及能量)的量。就装置和系统组件的布置而言,其也提供了完全不同的架构。

[0295] 该命名系统的优势在于,用户能够通过可记住词语(例如,公司名称)而不是哈希摘要标识元网中的顶级域。这也使搜索域更快,因为搜索关键字而不是哈希摘要更快。这也减小了输入错误,因此提供了对于区块链存储的数据的改进的搜索工具。

[0296] 鉴于具有从域名到节点索引的映射,我们可建立类似于因特网的统一资源定位符(URL)的资源定位符。我们将此称为元网URL(MURL),且呈如下形式

[0297] $MURL = 'mnp:' + '//domain\ name' + '/path' + '/file'$ 。

[0298] URL的每个的分量—协议、域名、路径及文件—已被映射到MURL的结构,从而使对象更具用户直观性,并且使其能够与因特网的现有结构集成。

[0299] 这假设每个节点具有与其公钥(地址)相关联的名称,该公钥在域树内的层级处是唯一的。该名称始终为给定节点的MURL的最右侧分量。如果树中的同一层级处的两个节点具有相同的名称,则其将具有相同的公钥,因此采用最新版本。

[0300] 下表给出了元网协议与因特网协议之间的类比:

[0301]	因特网	元网
	网站/文件	节点
	所有者	公钥 P_{node}
	IP 地址 (非唯一的)	节点索引 (唯一的) $ID_{node} = H(P_{node} TxID_{node})$
[0302]	域结构	节点树结构
	域名系统	从根节点名称映射到节点索引
	URL http://www.bobsblog.com/path/file	MURL mnp://bobsblog/path/file

[0303] 表:因特网协议和元网协议之间的类比概要

[0304] 搜索元网

[0305] 我们已经限定了示例性实施例的元网图结构,使得每个节点具有唯一的索引并且可以具有归属于其的名称。这允许使用MURL定位内容。为了也实现快速搜索功能,我们允许将附加的关键字归属于节点。

[0306] 节点的固定属性为索引及父节点的索引,可选属性为名称和关键字。

[0307] 节点属性

```
{
    index:  $H(P_{node} || TxID_{node})$ ;
    index of parent:  $H(P_{parent} || TxID_{parent})$ ; (NULL if orphan)
    name: 'bobsblog';
    kwd1: 'travel';
    kwd2: 'barbados';
    :
}
```

[0309] 在一个示例中,用于搜索元网的实用方法可为首先使用区块探测器在区块链中搜查(trawl),并通过元网标志标识所有交易,检查其是否为有效元网节点,并且若如此,则在数据库或其他存储资源中记录其索引和关键字。然后可以使用该数据库通过所需关键字有效地搜索节点。一旦通过所需关键字找到节点的索引,则其内容可以从区块探测器提取并且被查看。

[0310] 举例来说,考虑图14的分支 P_1 ,其中,对应于公钥 P_0 、 P_1 和 $P_{1,1}$ 的节点分别表示主页、主题页及子主题页。这些节点被给定的名称为‘bobsblog’、‘summer’和‘caribbean’,其属性如下所示:

[0311] 主页节点 P_0

MURL: mnp:// bobsblog

{

index: $H(P_0||TxID_0)$;

index of parent: *NULL*

[0312] name: ‘bobsblog’;

kwd1: ‘travel’;

kwd2: ‘barbados’;

:

}

[0313] 主题页节点 P_1

MURL: mnp:// bobsblog /summer

{

index: $H(P_1||TxID_1)$;

index of parent: $H(P_0||TxID_0)$;

[0314] name: ‘summer’;

kwd1: ‘travel’;

kwd2: ‘barbados’;

:

}

[0315] 子主题页节点 $P_{1,1}$

MURL: mnp:// bobsblog /summer/caribbean

{

index: $H(P_{1,1}||TxID_{1,1});$

[0316] index of parent: $H(P_1||TxID_1);$

name: 'caribbean';

kwd1: 'travel';

kwd2: 'barbados';

:

[0317] }

[0318] 在该示例中,叶节点 $P_{1,1,1}$ 、 $P_{1,1,2}$ 和 $P_{1,1,3}$ 分别被给定的名称为'beaches', 'nightlife' 和 'food',并用于存储单独的博客文章。完整的域结构在页后的图(diagram overleaf)上示出,包括与树中每个节点有关的MURL搜索路径。

[0319] 我们应注意,元网也可以通过将由节点交易存储的内容的哈希存储为附加的属性而并入内容可寻址的网络(CAN)。这意味着元网节点也可以被索引并且通过内容哈希被搜索。

[0320] 上文所描述的命名及寻址方法提供了优于现有技术的众多技术优势,包括:

[0321] 1. 公钥地址-系统使用与区块链相同的公钥-私钥对来分配节点地址。这意味着相同的密钥集合用于加密货币资金的管理及内容数据的许可两者。这提供了有效且安全的方案。

[0322] 2. 去中心化的域-域名的发布通过包括仅可由工作量证明生成的 $TxID_{node}$ 而完全去中心化。域名也可以并入实现所需域公钥的公平分布的人类可识别的公钥 P_{vanity} (虚荣地址)。再次,该方案提供了增强的效率及安全性。

[0323] 3. 图结构-命名及寻址架构指定可以从包括元网节点的区块链数据的子集构造的图。该设计使用有序结构将因特网的复杂性映射到区块链,使得区块链完全复制其功能性及可扩展性,同时保持安全性。

[0324] 浏览器钱包应用

[0325] 前面已经描述了,在元网协议中,所有数据均直接存在于区块链本身上。在本部分中,我们提供说明性计算机应用的实施例,其可以有效地访问、显示存储在区块链上的元网数据并与存储在区块链上的元网数据交互,本文中为方便起见仅称为“浏览器钱包”。

[0326] 我们将首先讨论浏览器钱包如何与分布式对等因特网接口的核心组件及功能性,之后在该部分的剩余部分中提供较详细的描述。

[0327] 概述

[0328] 组件

[0329] 浏览器钱包是旨在允许最终用户与区块链上的元网基础设施交互的应用。该应用应允许探索式搜索元网图以找到嵌入于树中的特定内容。另外,浏览器钱包将处理内容的检索、解密、重组及高速缓存(可选的)。

[0330] 浏览器钱包应用将通过支持本地(或外部)钱包而将这些元素与加密货币支付机制组合。浏览器钱包将包括被组合为单个计算机应用的以下核心元素。

[0331] 区块链搜索引擎-支持第三方搜索引擎通过各种索引(包括ID_{node}、节点名称、关键字、区块高度和TxID)来查询元网节点。

[0332] 显示窗口-解包(unpack)由全副本的区块链对等方返回到浏览器的内容的软件。这涵盖了访问令牌的解密、重组、高速缓存和兑换。

[0333] 加密货币钱包-区块链的货币的专用密钥管理。可以是应用本地的或被授权以与外部钱包(软件或硬件)通信及同步。能够写入标准区块链交易以及新的元网节点交易。可调解访问密钥及访问令牌的链上购买。

[0334] 分层的确定性密钥管理可同时用于加密货币公钥和元网节点地址。

[0335] 访问密钥/令牌钱包-用于购买的访问密钥或令牌的专用密钥管理。可以使用加密货币钱包接收购买的密钥或令牌,但不具有对密钥或令牌的许可。密钥或令牌可对用户隐藏以允许稍后到期。这可以通过使用受信任的执行环境来实现。可通过与区块链同步并查询当前区块高度来保证定时访问。

[0336] 功能性

[0337] 元网浏览器钱包的规范确保应用的以下功能。

[0338] 1. 分层密钥管理-用于控制资金及管理元网树(图)的密钥利用相同的分层确定性密钥基础设施,从而减小用户维护其元网内容的密钥记录的负担。

[0339] 2. 指向外部加密货币钱包-授权及与外部(非应用本地)钱包同步的能力通过移除作为故障点的浏览器钱包而允许附加的安全性。

[0340] 该应用可以写入区块链交易,并需要容纳密钥的外部钱包的签名,从而将这种责任委托给单独的软件或硬件。

[0341] 3. 元网内容的搜索-浏览器钱包可以支持和查询第三方搜索引擎,该引擎的功能可以包括爬网、编索引、服务及评级全局数据库中的元网节点交易数据。可以构造包含元网协议标志的OP_RETURN交易的数据库。参见BitDB 2.0-<https://bitdb.network/>。

[0342] 搜索引擎可以为浏览器钱包提供节点索引,这允许找到数据。

[0343] 4. 读写数据到区块链-除了使用搜索引擎和全节点为浏览器提供内容外,对加密货币钱包的支持还允许将内容直接从浏览器钱包写入元网。

[0344] 5. 数据的解压缩和解密-浏览器钱包处理解密密钥,并且可以原位对元网内容执行解压缩。

[0345] 6. 高速缓存节点身份(ID_{node})-可以在本地高速缓存唯一的节点身份,以便更高效地进行查找和查询。

[0346] 7. 旁路网络服务器-在给定节点索引的情况下,浏览器钱包可查询点对点(P2P)区块链网络的任何全副本成员以获取位于节点处的内容。因为元网存在于链上,所以任何全副本对等方必须具有节点及其内容的本地副本。

[0347] 这意味着用户的浏览器钱包只需要查询单个对等方,这可以直接完成而无需中间网络服务器。

[0348] 图15示出了浏览器钱包的示意图,以及如何在应用的不同组件之间分割其核心功能。

[0349] 区块链搜索引擎

[0350] 搜索引擎-现有技术

[0351] 现有技术中已知的搜索引擎(SE)依赖于强大的网络爬虫(crawlers)以根据用户查询来对网络内容进行定位、编索引及评级。(相同的基本原理可以扩展至对元网进行爬网的第三方区块链SE)。

[0352] SE通过查询中的关键字搜索标识相关的HTML元标签及内容。随后对爬网结果进行索引,其中,分析及编录任何嵌入的图像/视频/媒体文件。然后在考虑用户的位置、语言及装置的情况下,以编程方式对索引中最相关的结果进行评级。

[0353] 典型的SE应具有以下功能性:

[0354] 1.爬网-标识因特网数据并通过诸如域名、链接页面及相关关键字等相关元数据进行爬网。通过现有的内容发现新的因特网内容并且也针对任何相关信息进行爬网。

[0355] 2.编索引-分析并编录内容数据。该信息存储在数据库中。

[0356] 3.服务和评级-以与用户查询的相关性的顺序评级内容索引。

[0357] 区块探测器

[0358] 最接近因特网搜索引擎(SE)的区块链类似物是区块链探测器,其有时被称作‘区块探测器’或‘区块链浏览器’。区块链探测器是使能够在高层级上对区块链进行用户友好查询的网络应用,并且类似于网络浏览器起作用,但连接到区块链而非因特网。参见https://en.bitcoin.it/wiki/Block_chain_browser。

[0359] 在大多数情况下,这些探测器允许将区块(由区块标头的哈希进行索引)、交易(由TxID进行索引)、地址及未花费交易输出(UTXO)作为输入并对其进行搜索。许多探测器也提供其自身的应用编程接口(API)以用于检索原始交易及区块数据。参见<https://blockexplorer.com/api-ref>。

[0360] 区块探测器虽然能力不同,但大体上对于以用户易于摘录的形式编录交易并显示其基本信息—例如,交易的货币值、币的确认和历史以及地址—是有用的。许多探测器(例如,Bitcoin.com <https://explorer.bitcoin.com/bch>和Blockchain.com <https://www.blockchain.com/explorer>)也允许查看交易的单个输入及锁定脚本,尽管这些探测器与更高级的站点(例如,Blockchair <https://blockchair.com/>)选择如何提供该信息之间存在不一致。

[0361] 近来,用于基于区块链数据运行网络应用的基本区块链探测器存在许多扩展。这些应用(例如,Memo.cash <https://memo.cash/protocol>和Matter <https://www.mttr.app/home>)如区块探测器般对包含特定协议标识符的区块链交易进行编录及组织,以及显示那些特定交易内编码的数据。

[0362] 然而,使用区块链探测器存在两个重要问题,本发明的实施例解决了这些问题:

[0363] 1.通用性-当前不存在用于浏览存储在交易中的内容数据的行业标准。内容数据是指不涉及用于创建及保证基础区块链的协议的任何数据。

[0364] 2.关键字搜索-存储在交易中的内容数据需要可由人类可读的关键字检索。这通常不是当前的区块探测器的功能,因为当前的区块探测器用于查询交易的基于协议的性质(例如,区块高度、TxID及地址),而非将关键字作为搜索输入。(然而,如果词语直接包括在交易的脚本中,则例如Blockchair等一些探测器可搜索这些词语)。

[0365] 重要的是,如上所述,本发明的强大命名及寻址结构促进且实现了构造与本领域中所已知的相比更复杂的区块链探测器。

[0366] 提出的元网搜索引擎

[0367] 浏览器钱包应用与第三方搜索引擎通信以发现节点身份(ID_{node})。应设想到,该第三方可提供复制现有的因特网搜索引擎的能力的强大且多功能服务。

[0368] 元网搜索引擎第三方维护挖掘至区块链中的可由元网协议标志标识的所有元网交易的数据库。该数据库可以通过包括ID_{node}、节点名称、关键词、TxID及区块高度的范围索引编录所有元网节点。

[0369] 已经存在诸如Bit DB <https://bitdb.network/>等服务,其连续地与区块链同步且以标准数据库格式维护交易数据。浏览器钱包将对元网交易进行爬网、编索引、服务及评级的责任转给该第三方,并在定位存储在元网图上的内容时连接至其服务。

[0370] 通过具有仅专用于元网数据的数据库,可节约效率。不同于Bit DB,该数据库将不存储与所有交易相关联的数据,而仅存储包含元网标志的那些数据。诸如如MongoDB的非关系数据库等某些数据库在存储元网的图结构时可能更有效。这将允许更快查询、更低存储空间,及更有效地关联元网域内的相关内容。

[0371] 图16示出了当用户搜索元网基础设施内的内容时,浏览器钱包如何与第三方搜索引擎进行交互。重要的是,应注意,与因特网对比,无需路由,因此本发明在效率、速度、处理及所需资源方面提供重要优势。

[0372] 过程如下:

[0373] 1.最终用户在浏览器钱包搜索栏中输入关键字。

[0374] 2.浏览器钱包将关键字查询发送给第三方SE。

[0375] 3.SE针对其数据库检查关键字并返回包含相关内容的任何元网节点的ID_{node}。第三方也可向用户返回每个节点上的其他索引,以及提供对相关内容的建议。

[0376] 4.浏览器钱包使用节点标识和与其关联的域名来构造MURL。

[0377] 5.浏览器钱包从具有区块链全副本的任何网络对等方请求属于指定节点的内容。

[0378] 6.网络对等方将请求的内容提供给浏览器钱包。因为对等方具有区块链的副本,所以其必定也具有内容的副本,因此仅作出一个请求,且从不将请求转发给其他网络对等方。

[0379] 要强调的是,第三方SE仅负责对元网节点的属性记录进行索引并对其进行维护,而存储于节点上的原始内容数据则由具有区块链的全副本的网络对等方(例如,全副本对等方、矿工、档案馆)存储。

[0380] 内容显示-元网浏览器

[0381] 浏览器钱包应用模拟任何典型的网络浏览器都应提供的相同的前端能力。这些功能包括但不限于:

[0382] 1.搜索-提供对搜索引擎(SE)的访问权以定位内容。

[0383] 2.检索-与服务器通信以促进使用已知协议(例如,超文本传输协议(HTTP))传送内容。

[0384] 3.解译-解析原始代码(例如,以JavaScript)并执行。

[0385] 4.渲染-高效显示已解析的内容,供最终用户查看。

[0386] 5. 用户界面 (UI) - 为用户提供与内容进行交互的直观界面, 包括动作按钮和用于用户输入的机制。

[0387] 6. 存储 - 用于高速缓存因特网内容、cookies 等的本地临时存储容量, 以改善对内容的重复访问。

[0388] 在某些实施例中, 负责充当网络浏览器的浏览器钱包应用的软件组件能够对嵌入在区块链中的元网内容执行上述功能, 该元网内容是使用其属性可搜索的 (使用 SE) 并且可检索的 (从对等方)。

[0389] 重组、解压缩和解密

[0390] 根据本发明的某些实施例, 浏览器钱包应用的网络浏览器软件组件能够处理需要对给定元网内容执行的所有操作。大体而言, 存在需要执行的许多这样的操作, 但我们假定至少以下操作由应用使用元网协议及基础设施执行。

[0391] 重组 - 在元网内容需要被分割并插入到多个单独的节点交易中的情况下, 应用将从所有相关节点请求内容并重构原始内容。可以使用每个节点的属性中的附加标志编码碎片化内容的排序及结构。

[0392] 解压缩 - 在内容数据以压缩形式存储在区块链上的情况下, 应包括向浏览器钱包指示已使用哪个标准压缩方案的标志。应用将根据该标志解压缩内容。

[0393] 解密 - 在内容被加密的情况下, 应使用标志来表示加密方案。应用将从其解密密钥钱包 (如下文所讨论的) 定位密钥, 并根据使用的加密方案解密内容数据以供使用。

[0394] 在对内容数据执行这些操作时, 标志可用于向浏览器钱包表示需要执行给定操作。这适用于任何其他操作, 为此合适的 <operation_flag> 可以被包括为该操作所应用的节点的属性的一部分。

[0395] 高速缓存

[0396] 本地文件及 cookies 的高速缓存是典型网络浏览器的共同且重要的功能。浏览器钱包应用也以类似的方式使用本地存储, 以便可选地保存涉及关注内容的 ID_{node} 及其他节点属性的记录。这允许从频繁访问的元网节点更有效地查找并检索内容。

[0397] 元网解决了高速缓存因特网数据的固有问题, 该因特网数据是可变的且可由网络浏览软件根据提供商进行改变或删减。在高速缓存元网数据时, 用户可始终容易地验证数据与最初作为不可变记录包括在区块链上时处于同一状态。

[0398] 加密货币钱包

[0399] 分层确定性密钥管理

[0400] 确定性密钥 Dk 是从单个“种子”密钥初始化的私钥 (请参见, Andreas M. Antonopoulos, “Mastering Bitcoin (精通比特币) 中的第 5 章, O’Reilly 第 2 版, 2017 年, 第 93-98 页”)。种子是充当主密钥的随机生成的数字。哈希函数可用于将种子与其他数据 (例如, 索引号或“链码”) 组合 (请参见, HD 钱包 - BIP-32/BIP-44), 以得出确定性密钥。这些密钥彼此相关, 并且可以通过种子密钥完全恢复。如果用户希望将外部钱包与元网浏览器钱包结合使用, 则该种子还允许在不同的钱包实现之间轻松导入/导出钱包, 从而给予附加的自由度。

[0401] 分层确定性 (HD) 钱包是公知的确定性密钥的衍生方法。在 HD 钱包中, 父代密钥生成一系列子代密钥, 这些子代密钥又会衍生一系列孙密钥, 依此类推。这种树状结构是用于

管理若干密钥的强大机制。

[0402] 在优选实施例中,HD钱包可以被合并到图16中所示的元网架构中。

[0403] 使用HD钱包的优势包括:

[0404] 1. 结构可出于不同的目的使用子密钥的不同分支表达附加的组织含义。例如,用户可将不同分支(及其对应的子密钥)专用于不同类型的数据。

[0405] 2. 安全性用户可以在没有对应的私钥的情况下创建一系列公钥,从而使HD钱包具有仅接收能力的功能,并且适于在不安全服务器上使用。而且,由于需要存储较少的秘密,因此暴露风险较低。

[0406] 3. 恢复如果密钥丢失/损坏,则可以从种子密钥中将其恢复。

[0407] 本地(内部)和外部钱包支持

[0408] 有利地,本发明的实施例可直接合并常规的网络浏览器与一或多个加密货币钱包的功能性。从根本上而言,这是元网如何将对于“因特网”内容的支付与其向最终用户的交付组合。

[0409] 为实现这一点,浏览器钱包的实施例可具有用作加密货币钱包的专用内置软件组件。该钱包是应用自身本地的且可用于管理加密货币私钥,并授权作为对于浏览器钱包自身内的元网内容支付的交易。

[0410] 这意味着应用的浏览器组件可提示钱包组件授权所需的支付—通过购买解密密钥、访问令牌或其他方式—以查看元网内容。应用无需调用外部第三方来处理支付,因此关注的元网内容由应用消费并就地支付。

[0411] 外部钱包

[0412] 如果用户希望在外币钱包(软件或硬件)上管理或保存其加密货币私钥或甚至使用多个钱包,则可通过应用的实施例实现相同的优势及功能性。这可代替或结合应用的本地钱包执行。

[0413] 在这样的实施例中,应用建立与外部钱包的链接或配对并与之同步,但并不在浏览器钱包自身中存储私钥。而是,当浏览器组件提示为内容进行支付时,应用从所选的外部钱包中请求通过数字签名进行授权。该授权由用户作出,并且浏览器钱包可广播交易并查看支付的内容。

[0414] 读取及写入元网交易

[0415] 元网的本质优势为其使用相同的数据结构—区块链—来记录支付及内容数据两者。这意味着除了创建仅基于加密货币的交换的交易之外,软件钱包还可用于将内容数据写入到元网基础设施。

[0416] 内置到应用的本地钱包能够将交易写入到区块链,与典型的简化支付验证(SPV)客户端相比,这些交易更复杂—请参见<https://bitcoin.org/en/glossary/simplified-payment-verification>。钱包允许用户通过从其计算机选择待嵌入区块链中的内容数据而选择将元网节点交易直接从应用写入到区块链中。

[0417] 由于浏览器钱包应用具有用户接口(UI),因此其允许钱包组件创建并广播包括内容数据的交易,该内容数据已预先在浏览器组件中或用户计算机上被构造。对于自行进行处理的专门设置的钱包而言,要实现此功能将非常困难。

[0418] 访问密钥/令牌钱包

[0419] 前面已经描述了,内置于元网协议中的是使用ECC密钥对或AES对称密钥对内容进行加密的能力,以及购买对应的解密密钥或令牌的能力。我们将这些密钥或令牌称为访问密钥或访问令牌。

[0420] 这些密钥/令牌授予用户查看或编辑内容的许可(单次使用或多实例使用),并且与控制用户加密货币钱包的密钥起着不同的作用(尽管在需要时同一密钥可用于这两目的)。出于此原因,有利的是引入与应用的本地加密货币钱包分离的用于存储及管理访问密钥及令牌的新钱包。

[0421] 也可以通过允许访问密钥/令牌在某一时间段之后被烧毁而引入对元网内容的定时访问的概念。这可以通过以下方式来实现:要求访问密钥/令牌存储在受信任的执行环境(TEE)中且用户不可直接访问访问密钥/令牌。

[0422] 访问密钥/令牌可以被“烧毁”的事实也是不将其存储在加密货币钱包中以确保不存在加密货币私钥被烧毁的风险的动机因素。

[0423] 以类似于加密货币钱包的方式,解密密钥及访问令牌可以确定性地被存储及管理以促进有效处理和部署。可以通过对主钥的后续添加而生成并恢复解密密钥(例如,ECC私钥),而可以使用由一些初始令牌播种的哈希链重构访问令牌。

[0424] 此处重要的是,区分加密货币钱包处理密钥对的确定性密钥生成,该密钥对用于与其他用户进行交易并创建新的元网节点,而密钥/令牌钱包处理已由加密货币钱包购买的密钥及令牌。

[0425] 区块高度许可

[0426] 时间锁可以包括在比特币脚本语言中,以实现区块高度许可。`op_code OP_CHECKLOCKTIMEVERIFY (CLTV)` 设置允许花费交易输出(UTXO)的区块高度。

[0427] 区块高度许可的优势是双重的:

[0428] 1.版本控制-在元网协议中,节点的最新版本可以从最大区块高度处的节点标识。浏览器钱包可设置成通过区块高度仅显示文件的最近版本,从而实现工作量证明版本控制。

[0429] 2.定时访问-浏览器钱包应用可周期性地烧毁由用户在原子级上购买的解密密钥。这确保了查看者仅可在其已进行支付的时间段期间访问内容数据。可以通过将解密密钥存储在受信任的执行环境(TEE)中来防止解密密钥的克隆。此外,原子交换涉及确定性密钥Dk(用于解密内容数据)的购买。尽管该确定性密钥是公开可见的,但是TEE可用于签署Dk与安全隔区的(securely enclaved)私钥的组合。

[0430] 浏览器钱包可以被布置成与区块链的当前状态同步,以便将区块高度用作其自身的时间代理,而不是依赖任何外部时钟或第三方时间预告。

[0431] 旁路网络服务器

[0432] 本发明允许浏览器(客户端)和网络服务器通过旁路域名系统(DNS)服务器和典型的网络路由程序的分布式对等互联网通信和交换信息的新机制。参见http://www.theshulers.com/whitepapers/internet_whitepaper/。本发明提供了包括维护区块链的全副本的对等方的新网络架构,从该新网络架构可以向浏览器钱包应用提供内容。

[0433] 本地全副本对等方

[0434] 考虑在每个地理区域(例如,邮区、城镇、城市)中的本地对等方的系统。我们假定

在该局域网内,至少一个对等方维护区块链的全副本,我们将该对等方称为本地全副本对等方(LFCP)。出于我们的目的,LFCP仅需要存储包括元网标志的区块链交易,但不将其限于此。

[0435] 所有用户默认向LFCP发送‘获得’请求。由于对等方维护整个区块链的完整且最新副本,因此所有请求都可以得到满足,因为所查询的任何节点ID对于LFCP而言将是可用的。应注意,如果SE足够强大且足够大到能够存储元网内容并执行典型SE的主要功能,则元网搜索引擎也可以充当LFCP。

[0436] 在最简单的情况下,每个LFCP将具有相同的存储及磁盘空间开销,因为其都将需要能够存储全部区块链(在写入时约200GB)。每个LFCP之间的区别为LFCP应扩展其能力以对来自元网用户的本地请求的要求作出响应。因此,如果默认世界上每个元网用户均查询其最接近的LFCP,则每个LFCP均应力求扩展其操作能力以满足其本地要求。如城市等人口密集区域将需要包括许多集群服务器的LFCP操作,而如小镇等人口稀少区域将需要较少的LFCP操作。

[0437] 重要的是要注意,磁盘空间要求是通用的,而对于每个LFCP的CPU要求适应于局域网需求。这是适应性网络的示例,例如,Freenet—参见<https://blockstack.org/papers/>。

[0438] 这样的系统的一个优势为在检索与给定ID_{node}相关联的内容时,用户仅需要单次(本地)连接至其LFCP。LFCP无需将请求转发至其他对等方,因为其自身保证能够提供所需内容。

[0439] 元网提供优于因特网的许多优势—例如,去中心化及去除重复—类似于如IFPS等其他点对点(P2P)文件共享服务。然而,元网通过确保不可变性,及至关重要地通过去除使网络充斥对于给定内容的请求的需要而改进这些现有的P2P模型。

[0440] 元网基础设施也通过采用这些对等方的网络而对于任何一个LFCP的受损也是鲁棒的。这意味着,如果停用LFCP,则最终用户简单地默认使用其下一个最接近LFCP。如果LFCP彼此通信以在任何给定时间指示哪些附近的对等方就请求而言低于或高于容量,则这可以更有效。这可以允许用户将其请求发送到最适当的对等方并在附近的LFCP之间建立请求分配的动态平衡。

[0441] 全局全副本对等方

[0442] 现在考虑当通用磁盘空间要求变得对于较小的对等方而言过大时的场景,随着区块链的元网部分随采用而扩展和增长会发生该情况。

[0443] 在这种情况下,较小的LFCP应基于流行度系统来使用其磁盘空间容量存储元网节点交易(存在用于通过请求量及本质评级内容的现有技术)。这意味着LFCP现在修整其CPU(用于请求处理能力)及其存储分配(用于内容服务能力)两者,以适应其在内容量及本质两方面的本地地理需求。

[0444] 为了解决LFCP现在不能存储所有元网交易内容的事实,可以利用全局全副本对等方(GFCP)的概念。GFCP为具有以下性质的全副本对等方:

[0445] 1.GFCP增长其磁盘空间容量以便始终维护区块链的全副本。

[0446] 2.GFCP具有相当大的CPU资源,使得与LFCP相比,其可以处理明显更多的请求。万一许多LFCP受损,则全局全副本对等方应能够应对需求的突然增加。

[0447] GFCP存在两个主要功能。首先,在请求从LFCP溢出时,充当对于元网内容的用户请

求的故障保护。其次，GFCP充当存档对等方以存储历史上挖掘的所有元网内容，这确保了即使许多LFCP从其本地存储供给中省略了某些内容，仍可访问任何元网节点内容。

[0448] 全局数据库(data bank)

[0449] GFCP的概念是强大的且说明了元网的总架构如何提供对现有问题的解决方案；创建涵盖所有的全局数据库。

[0450] 在此之前，尚不可能安全地构造通用且可全局访问的数据库，因为需要由中心机构来维护数据库。该中央机构将故障点和信任点注入到系统中。至关重要地，如果依赖于一个组织来存储并维护所有因特网数据，则需要相信该组织正确且合法地操作，而不会损毁信息的事实。

[0451] 通过元网基础设施，有效地从全局数据中心的概念中移除了信任及中心性这两个问题。现在，可创建这样的GFCP，因为仅依赖于其来提供存储所需的磁盘空间而不验证及认证待存储的信息。

[0452] 通过元网，验证存储内容的过程由矿工进行，因此通用全局数据库可以是受信任的，因为其无法破坏区块链信息。GFCP无需是受信任的，仅需要提供存储。

[0453] 所有GFCP可以存储针对区块链自身始终是可验证的且可证明的相同信息的事实意味着可以在许多这样的GFCP之间复制信息。

[0454] 这意味着通过使许多全局数据库并行存在且可证明地存储相同信息也解决了具有单个故障点的问题。

[0455] 图17示出了具有两个LFCP和一个GFCP的系统，且说明了每个对等方可以如何在对于各个对等方的损坏是鲁棒的网络中支持另一对等方。

[0456] 可以在上述浏览器钱包应用的实施例中实现的本发明的各方面提供了许多优于现有技术的区别特征和优势，包括但不限于：

[0457] 1. 确定性密钥-在应用的同一钱包组件中执行用于加密货币及元网地址两者的分层确定性密钥管理。这允许通过减小其存储要求且实现密钥恢复的多个功能组织密钥。

[0458] 2. 支付机制-该应用允许消费者直接向商家支付，而无需指向将按照惯例认证并提供信任的另一个应用或第三方支付服务。这允许经由同一区块链平台进行数字内容的购买及交付。该应用继承比特币支付的优势，包括低值交换或涉及多方的更复杂的交易。

[0459] 3. 旁路网络服务器-该应用促进对将按照惯例处理大量流量、请求及路由的常规网络服务器的旁路。这是因为应用仅需要从单个LFCP请求内容，这保证了无需将请求转发至其他LFCP来服务用户。这减小了总流量以及每个请求的完成时间

[0460] 4. 定时的访问-该应用通过与区块链同步并基于其当前状态使用区块链来实施访问许可而促进对内容的定时访问。这移除了对随时间监测用户特权的第三方服务的需要，同时保护原始所有者的权利。

[0461] 用例-去中心化应用商店(Swapp商店)

[0462] 这里呈现的元网架构的第一用例(仅出于说明性目的)为对于应用(app)的去中心化支付及分布。

[0463] 考虑如下场景：app开发者爱丽丝与消费者鲍勃希望彼此交易。该交易将采用原子交换的形式，其中，以金钱交换授予鲍勃对应用数据的访问权限的秘密密钥。加密的应用数据已作为元网节点交易的一部分公开。

[0464] 原子级上交换的应用被称为Swapp。第三方平台 (Swapp商店) 可用于对存在于元网上的应用进行编录并通告,但访问密钥的支付及传送到用户 (例如,鲍勃) 不需要涉及任何第三方且可直接在商家与消费者之间进行。

[0465] 以下部分详述了从爱丽丝创建app至鲍勃部署该app的可用于购买及出售Swapp的过程。在整个过程中,爱丽丝及鲍勃将使用其各自的浏览器钱包与元网交互。

[0466] 发布

[0467] 1. 爱丽丝编写应用。构成该应用的数据是由<App>表示的内容。她还使用秘密密钥 S_k 将其加密为<e (App)>。

[0468] 2. 爱丽丝创建节点交易ID $_{AliceApp}$ 来设置她的第一元网域 (树)。她生成将用作节点地址的1AliceAppHtKNgkdXEobR76b53LEttyp ($P_{AliceApp}$)。

[0469] 3. 然后,爱丽丝创建第一节点的子代,以形成树,该树与她的应用的元网库相对应。爱丽丝的树域在图18中示出。

[0470] 该树上的叶节点之一是对应于具有索引ID $_{App}$ 的她的应用程序<App>的节点。在该节点中,爱丽丝将加密的应用数据<e (App)>插入到节点的输入脚本 (scriptSig) 中。使用秘密密钥 s_k 来使用Koblitz方法加密app数据。

[0471] 该节点交易如下所示。

	$TxID_{App}$	
[0472]	输入	输出
[0473]	<Sig P_{puzzle} > < P_{puzzle} > <e(App)> OP_DROP	OP_RETURN <Metanet Flag> < P_{App} > < $TxID_{puzzle}$ >

[0474] 4. 爱丽丝公开广播ID $_{AliceApp}$ 、 $P_{AliceApp}$ 和域名‘AliceApp’。这可以经由社交媒体、因特网网站或通过使用第三方元网网站来实现。

[0475] 购买

[0476] 1. 鲍勃想要下载益智游戏,并在其浏览器钱包上查看到的元网网站 (Swapp商店) 上看到列出的爱丽丝的app。

[0477] 2. 然后,鲍勃使用来自网站的信息与爱丽丝通信,并设置原子交换。交换被设计成使得鲍勃将以比特币向爱丽丝支付商定的价格,并且爱丽丝将揭露秘密密钥 s_k ,或这些事件均不发生。

[0478] 3. 原子交换完成,鲍勃的浏览器钱包将秘密密钥 s_k 存储在其访问密钥/令牌钱包中。

[0479] 部署

[0480] 鲍勃现在具有将允许其解密爱丽丝先前发布的应用数据的密钥 s_k 。为了下载app并对其进行部署,鲍勃进行以下操作。

[0481] 1. 鲍勃使用元网搜索引擎 (SE) 找到与加密的app数据<e (App)>相关联的MURL。其在浏览器钱包中将关键字‘AliceApp’和‘App’用作搜索栏的输入。第三方SE解析查询并返

回以下MURL:

[0482] <mnp://aliceapp/games/puzzle/app>

[0483] 该定位符对应于唯一的元网节点 ID_{App} ,该节点在其输入脚本中包括加密的app数据。

[0484] 2. 鲍勃的浏览器钱包接收该MURL,然后向最近的适当LFCP发送请求。该对等方为鲍勃提供请求的数据 $\langle e(App) \rangle$ 。

[0485] 3. 浏览器钱包根据 ID_{App} 的属性处理数据。这包括使用秘密密钥 s_k 解密应用数据并处理 $\langle App \rangle$ 。

[0486] 4. 鲍勃将应用 $\langle App \rangle$ 从他的浏览器下载到他的计算机。现在,鲍勃可以在本地部署应用,而不必重新购买访问权限。

[0487] 图19示出了以上说明性用例中概述的整个过程。流程图示出了两个动作分支:爱丽丝的分支(在左侧开始)及鲍勃的分支(在右侧开始)。对应于爱丽丝的分支示出初始发布阶段且鲍勃分支示出经由原子交换设置购买的阶段。

[0488] 在鲍勃的分支上,他广播了以下交易 $TxID_{Bob}$ 作为原子交换设置阶段:

$TxID_{Bob}$			
输入		输出	
值	脚本	值	脚本
x BCH	$\langle Sig P_B \rangle \langle P_B \rangle$	x BCH	[Private Key Puzzle P_k, r_0] [CheckSig P_A]

[0490] 在该交易中,输出通过私钥谜题锁定,该私钥谜题需要将秘密解密密钥 s_k 揭露给鲍勃以便爱丽丝花费。

[0491] 该图中爱丽丝和鲍勃的分支在爱丽丝成功地完成原子交换交易的点处汇聚。这在爱丽丝广播交易 $TxID_{Alice}$ 时实现:

$TxID_{Alice}$			
输入		输出	
值	脚本	值	脚本
x 比特币	$\langle Sig P_A \rangle \langle P_A \rangle \langle Sig P_1, r_0 \rangle \langle P_1 \rangle$	x 比特币	[CheckSig P_A]

[0493] 一旦该交易被广播,爱丽丝和鲍勃的动作分支就再次发散。爱丽丝接收 x 个比特币的支付,而鲍勃接收秘密解密密钥 s_k ,并且能够从元网检索和解密爱丽丝的应用。

[0494] 现在转向图20,提供了可用于实践本公开的至少一个实施例的计算装置2600的说明性简化框图。在各种实施例中,计算装置2600可以用于实现以上示出和描述的任何系统。例如,计算装置2600可以被配置为用作数据服务器、网络服务器、便携式计算装置、个人计算机或任何电子计算装置。如图20所示,计算装置2600可以包括具有一个或多个级别的高速缓冲存储器和存储器控制器的一个或多个处理器(统称为2602),该处理器可以被配置为与包括主存储器2608和永久存储装置2610的存储子系统2606通信。如图所示,主存储器

2608可以包括动态随机访问存储器 (DRAM) 2618和只读存储器 (ROM) 2620。存储子系统2606和高速缓存存储器2602可以用于存储信息,例如与本公开中所描述的与交易和区块相关联的详细内容。(一个或多个)处理器2602可用于提供本公开中所描述的任何实施例的步骤或功能。

[0495] (一个或多个)处理器2602还可以与一个或多个用户接口输入装置2612、一个或多个用户接口输出装置2614以及网络接口子系统2616通信。

[0496] 总线子系统2604可以提供用于使得计算装置2600的各个组件和子系统能够按预期彼此通信的机制。尽管总线子系统2604被示意性地示出为单个总线,但是总线子系统的替代实施例可以利用多个总线。

[0497] 网络接口子系统2616可以提供至其他计算装置和网络的接口。网络接口子系统2616可以用作从不同于计算装置2600的其他系统接收数据以及将数据传输到其他系统的接口。例如,网络接口子系统2616可以使数据技术人员能够将装置连接至网络,使得数据技术人员可以在位于远程位置(例如,数据中心)的同时,将数据传输到该装置并从该装置接收数据。

[0498] 用户接口输入装置2612可以包括一个或多个用户输入装置,例如,键盘;诸如集成鼠标、轨迹球、触摸板或图形输入板等定点装置;扫描仪;条形码扫描仪;合并到显示器中的触摸屏;诸如语音标识系统、麦克风等音频输入装置;以及其他类型的输入装置。通常,术语“输入装置”的使用旨在包括用于将信息输入到计算装置2600的所有可能类型的装置和机制。

[0499] 一个或多个用户接口输出装置2614可以包括显示子系统、打印机或诸如音频输出装置等非可视显示器。显示子系统可以是阴极射线管 (CRT)、诸如液晶显示器 (LCD)、发光二极管 (LED) 显示器或投影仪的平板装置或其他显示装置。通常,术语“输出装置”的使用旨在包括用于从计算装置2600输出信息的所有可能类型的装置和机制。一个或多个用户接口输出装置2614可以用于例如呈现用户界面以有助于用户与执行所描述的过程及其中的变型的应用的交互,当这样的交互是适当的时。

[0500] 存储子系统2606可以提供计算机可读存储介质,该计算机可读存储介质用于存储可以提供本公开的至少一个实施例的功能的基本编程和数据构造。当由一个或多个处理器执行时,应用程序(程序、代码模块、指令)可以提供本公开的一个或多个实施例的功能,并且可以被存储在存储子系统2606中。这些应用程序模块或指令可以由一个或多个处理器2602执行。另外,存储子系统2606可以提供用于存储根据本公开而使用的数据的存储库。例如,主存储器2608和高速缓存存储器2602可以为程序和数据提供易失性存储。永久存储装置2610可以为程序和数据提供永久性(非易失性)存储,并且可以包括闪存、一个或多个固态驱动器、一个或多个磁性硬盘驱动器、一个或多个具有相关的可移动介质的软盘驱动器、一个或多个具有相关的可移动介质的光驱(例如,CD-ROM或DVD或Blue-Ray)驱动器,以及其他类似的存储介质。这样的程序和数据可以包括用于执行如本公开中所描述的一个或多个实施例的步骤的程序以及与本公开中所描述的交易和区块相关联的数据。

[0501] 计算装置2600可以是各种类型,包括便携式计算机装置、平板计算机、工作站或以下描述的任何其他装置。另外,计算装置2600可以包括可以通过一个或多个端口(例如,USB、耳机插孔、闪电连接器等)连接到计算装置2600的另一装置。可以连接到计算装置2600

的装置可以包括被配置为接受光纤连接器的多个端口。因此,该装置可以被配置为将光信号转换为电信号,该电信号可以通过将装置连接至计算装置2600的端口传输以进行处理。由于计算机和网络不断变化的性质,图20中所描绘的计算装置2600的描述仅旨在作为特定示例用于说明该装置的优选实施例的目的。具有比图20中描绘的系统更多或更少的组件的许多其他配置是可能的。

[0502] 应当注意,上述实施例说明而不是限制本发明,并且本领域技术人员将能够设计许多替代实施例,而不脱离由所附权利要求书限定的本发明的范围。在权利要求书中,括号中的任何附图标记都不应解释为对权利要求的限制。单词“包括 (comprising,comprise)”等不排除任何权利要求或整个说明书中列出的元素或步骤之外的元素或步骤的存在。在本说明书中,“包括 (comprise)”是指“包括 (include) 或由……组成 (consist of)”,“包括 (comprising)”是指“包括 (including) 或由……组成 (consisting of)”。元素的单数形式并不排除此类元素的复数形式,反之亦然。本发明可以通过包括几个不同元件的硬件以及通过适当编程的计算机来实现。在列举几个部件的装置权利要求中,这些部件中的几个可以由一个且相同的硬件来实施。在互不相同的从属权利要求中记载某些手段的事实并不表示不能有利地使用这些手段的组合。

TxID	
输入	输出
<Sig P> <P>	OP_RETURN <Metanet Flag> <Attribute 1> <Attribute 2>
	<Content 1> OP_DROP <H(P')> [CheckSig P']

图1

TxID	
输入	输出
<Sig P> <P> <Content 1> OP_DROP	OP_RETURN <Metanet Flag> <Attribute 1> <Attribute 2>

图2

$TxID_1$	
输入	输出
$\langle Sig P \rangle \langle P \rangle$	OP_RETURN $\langle Metanet Flag \rangle \langle Attribute 1,1 = Content name \rangle$ $\langle Attribute 1,2 = Recombination scheme \rangle$ $\langle Attribute 1,3 = Content chunk index 1 \rangle$
	$\langle Content chunk 1 \rangle OP_DROP \langle H(P') \rangle [CheckSig P']$

$TxID_2$	
输入	输出
$\langle Sig P \rangle \langle P \rangle$	OP_RETURN $\langle Metanet Flag \rangle \langle Attribute 2,1 = Content name \rangle$ $\langle Attribute 2,2 = Recombination scheme \rangle$ $\langle Attribute 2,3 = Content chunk index 2 \rangle$
	$\langle Content chunk 2 \rangle OP_DROP \langle H(P') \rangle [CheckSig P']$

图3

$TxID_{Bob}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_B \rangle \langle P_B \rangle$	x	$[Private Key Puzzle P_1, r_0] [CheckSig P_A]$

图4

$TxID_{Alice}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_A \rangle \langle P_A \rangle \langle Sig P_1, r_0 \rangle \langle P_1 \rangle$	x	$[CheckSig P_A]$

图5

使用		爱丽丝	鲍勃
令牌发布	10个令牌的购买	$I_{Alice} = k$	$I_{Bob} = H^{10}(Y)$
令牌兑换	兑换令牌1	X_1	$T_1 = H^9(Y)$
	兑换令牌2	X_2	$T_2 = H^8(Y)$
	...	...	...
	兑换令牌10	X_{10}	$T_{10} = Y$

图6

$TxID_{Alice}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_A \rangle \langle P_A \rangle$	x	$[Hash\ Puzzle\ H(I_{Alice})] [Hash\ Puzzle\ H(I_{Bob})] [CheckSig P_B]$

阶段 1.1 (爱丽丝到鲍勃) :

图7

$TxID_{Bob}$			
输入		输出	
值	脚本	值	脚本
$10 + x$	$\langle Sig P_B \rangle \langle P_B \rangle$	$10 + x$	$[Hash\ Puzzle\ H(I_{Alice})] [Hash\ Puzzle\ H(I_{Bob})]$ $[CheckSig P_A]$

阶段 1.2 (鲍勃到爱丽丝) :

图8

阶段 2.1 (鲍勃到爱丽丝) :

$TxID_{Bob}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_B \rangle \langle P_B \rangle$	x	$[Hash\ Puzzle\ H(X_1)] [Hash\ Puzzle\ H(T_1)] [CheckSig P_A]$

图9

阶段 2.2 (爱丽丝到鲍勃)

$TxID_{Alice}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_A \rangle \langle P_A \rangle$	x	$[Hash\ Puzzle\ H(X_1)] [Hash\ Puzzle\ H(T_1)] [CheckSig P_B]$

图10

阶段 2.3 (爱丽丝到爱丽丝) :

$TxID_{Alice}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_A \rangle \langle P_A \rangle \langle T_1 \rangle \langle X_1 \rangle$	x	$[CheckSig P_A]$

图11

阶段 2.4 (鲍勃到鲍勃) :

$TxID_{Bob}$			
输入		输出	
值	脚本	值	脚本
x	$\langle Sig P_B \rangle \langle P_B \rangle \langle T_1 \rangle \langle X_1 \rangle$	x	$[CheckSig P_B]$

图12

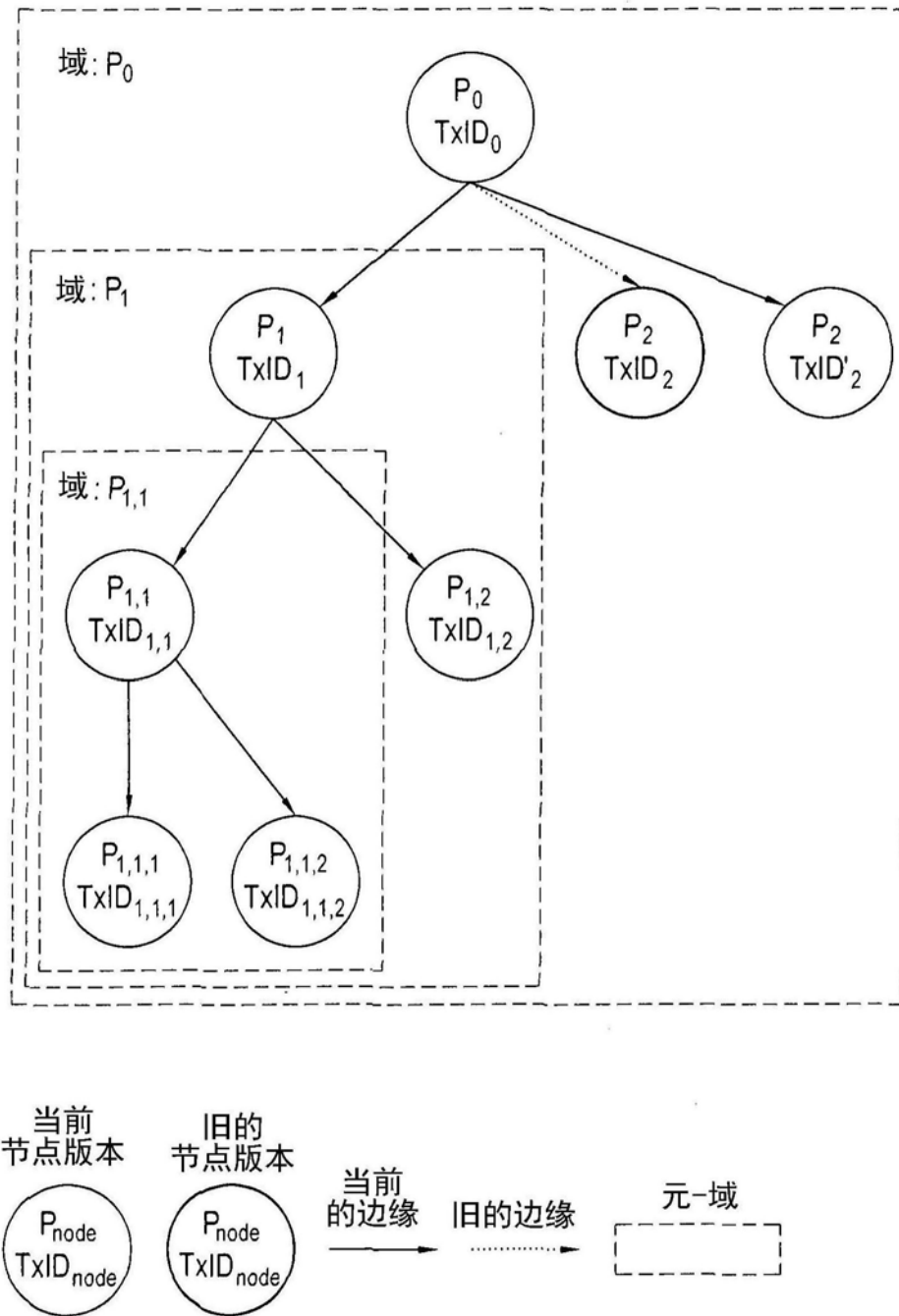


图13

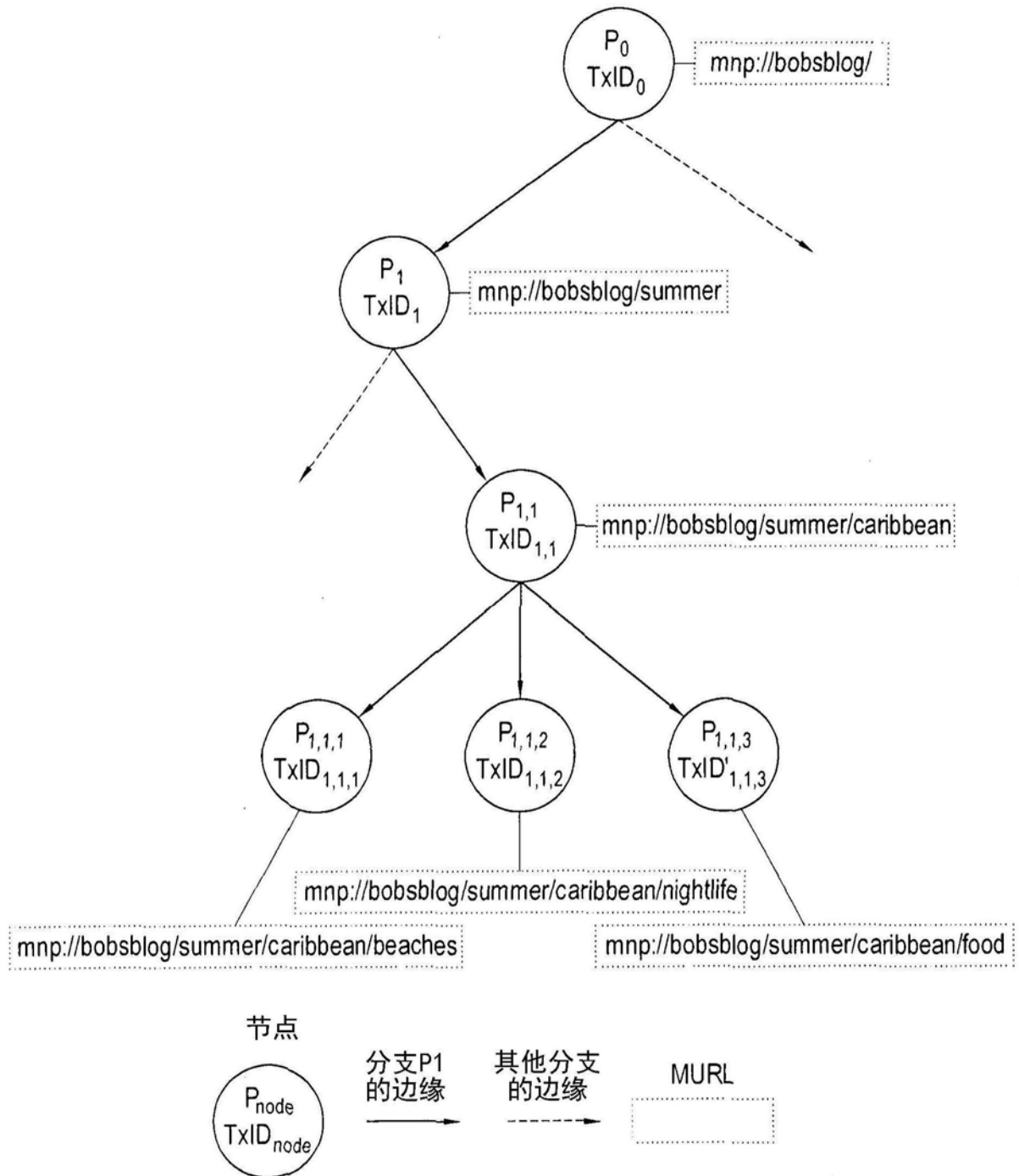


图14

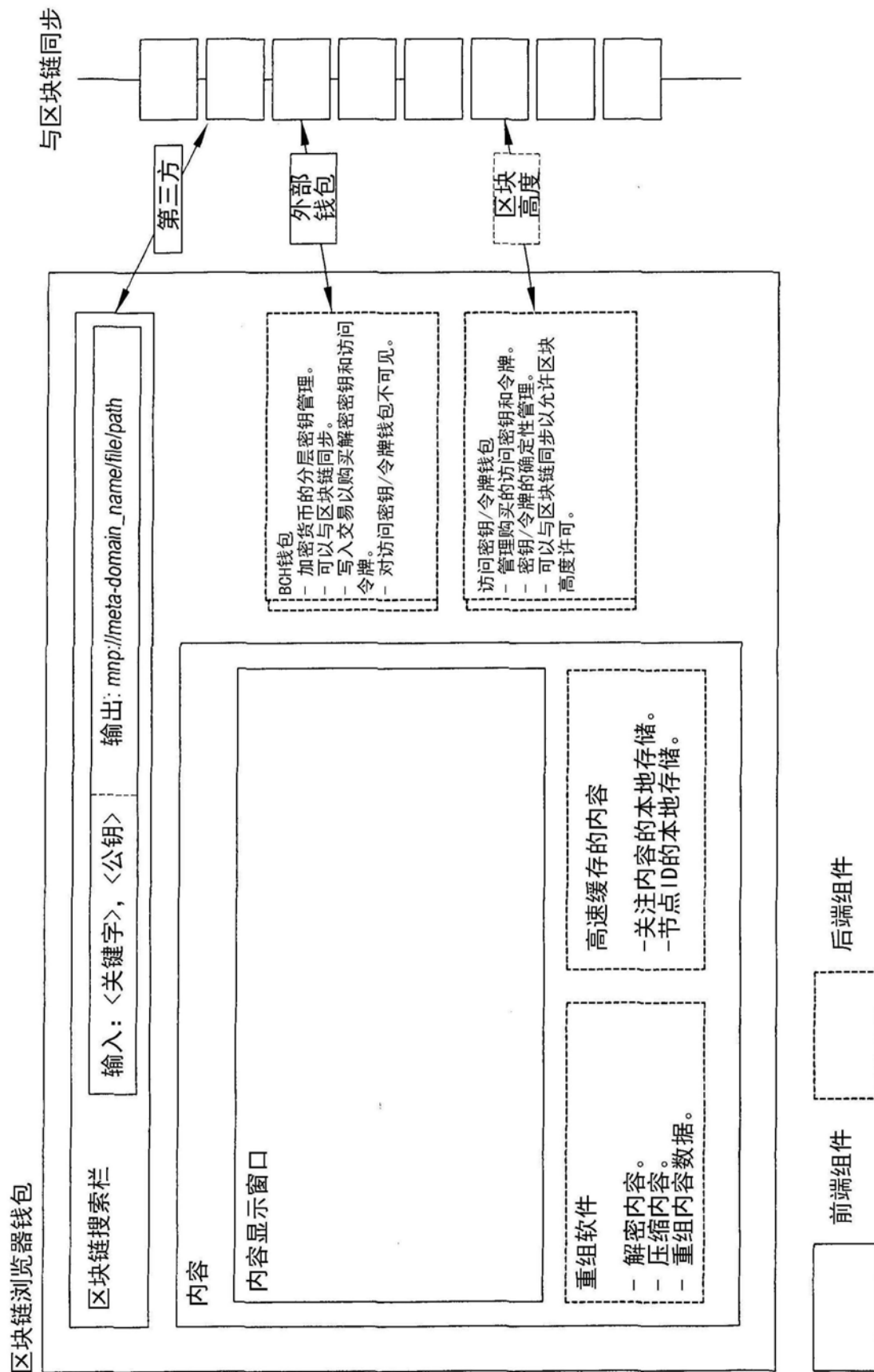


图15

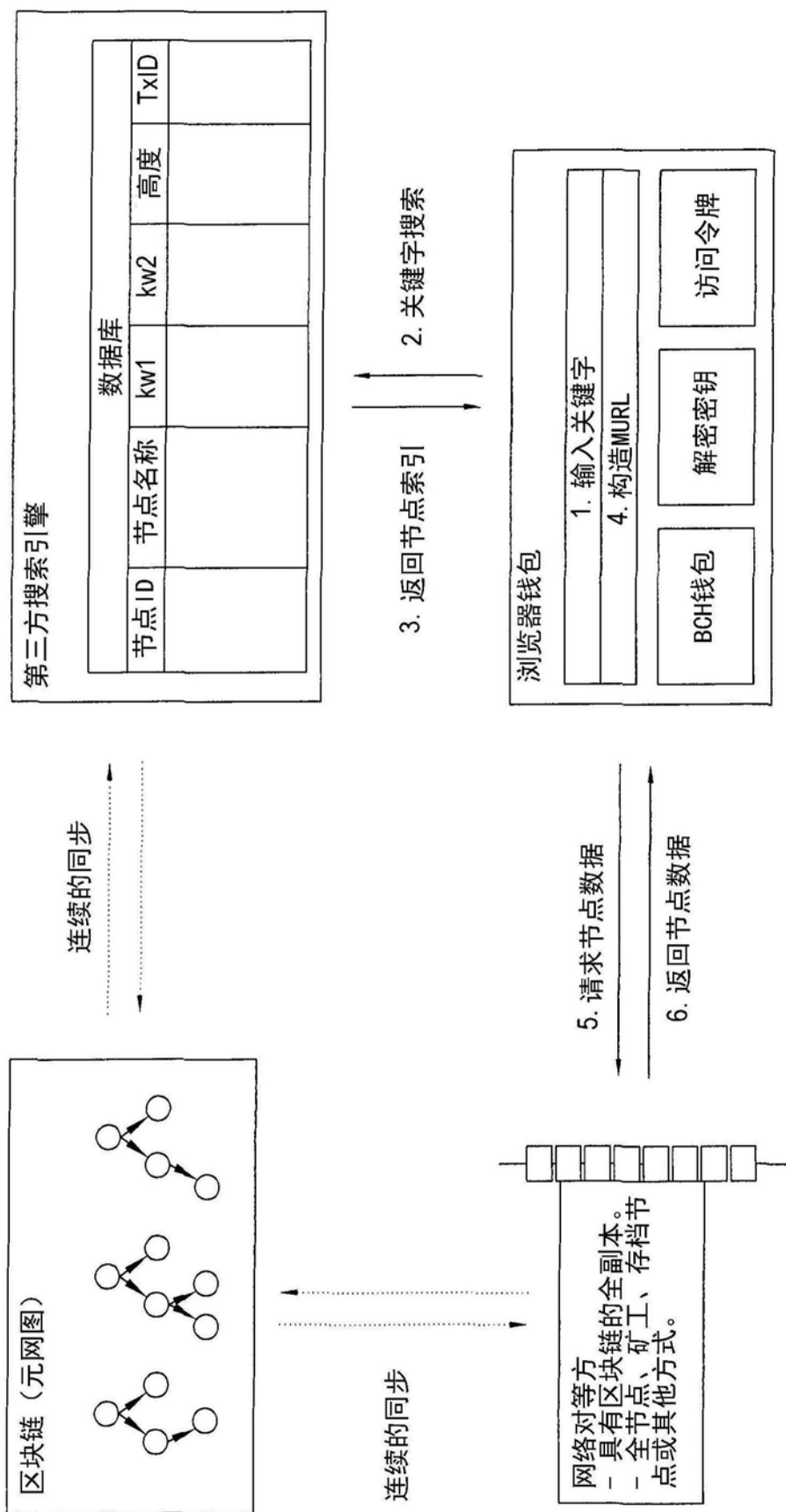


图16

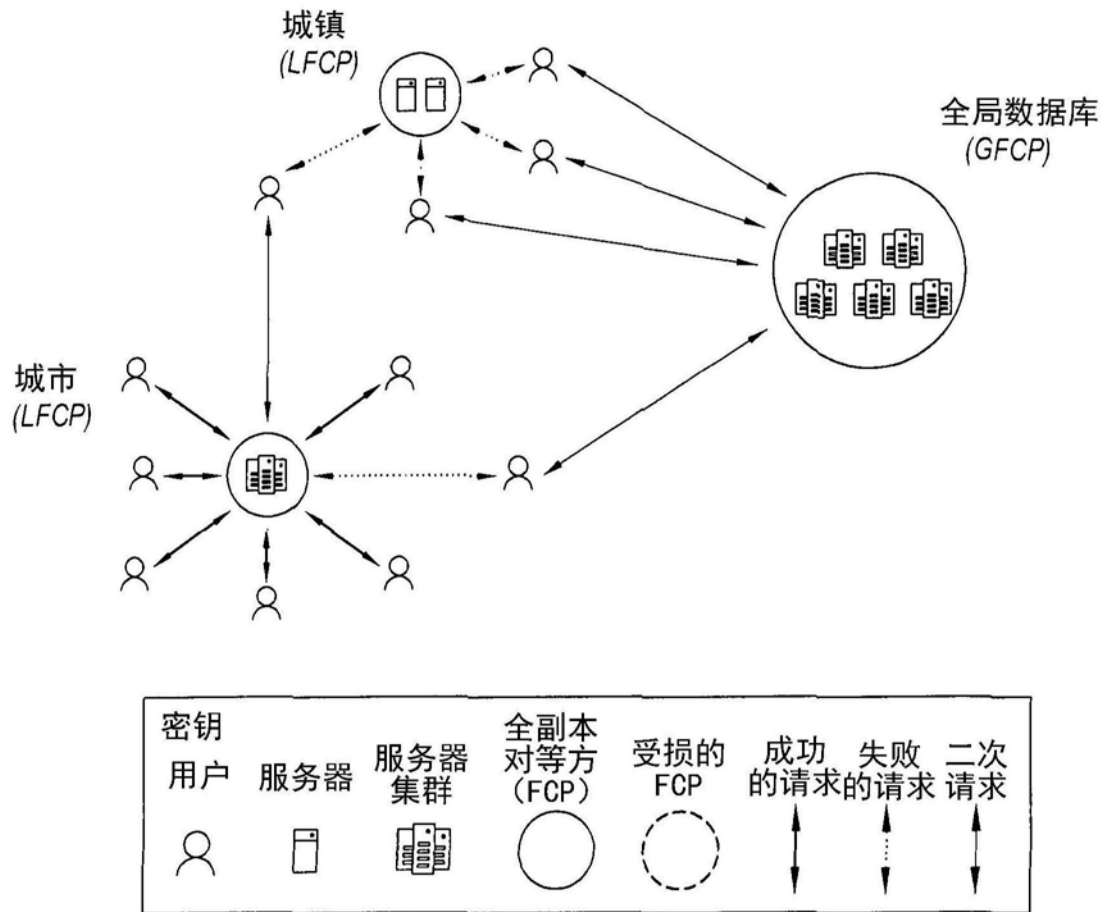


图17

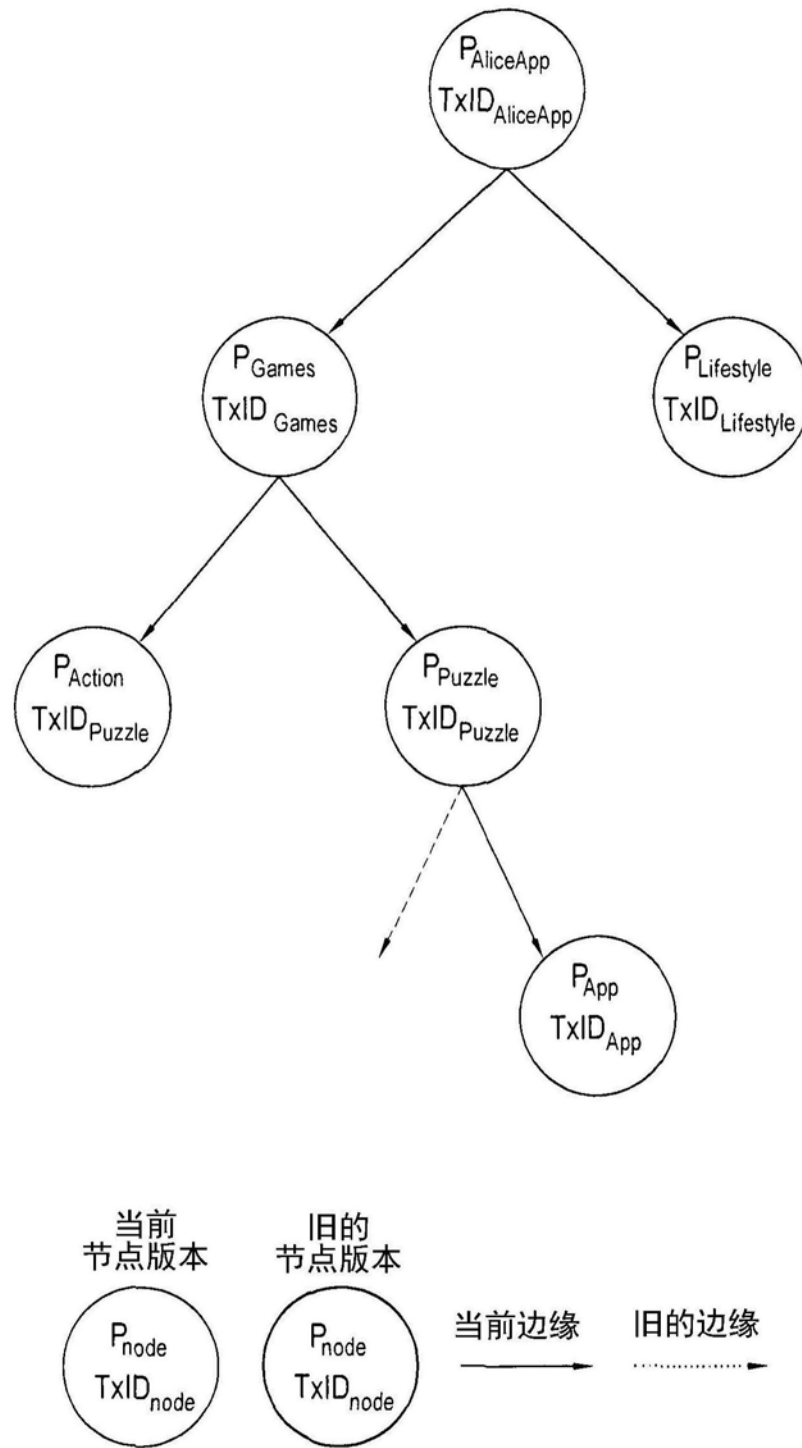


图18

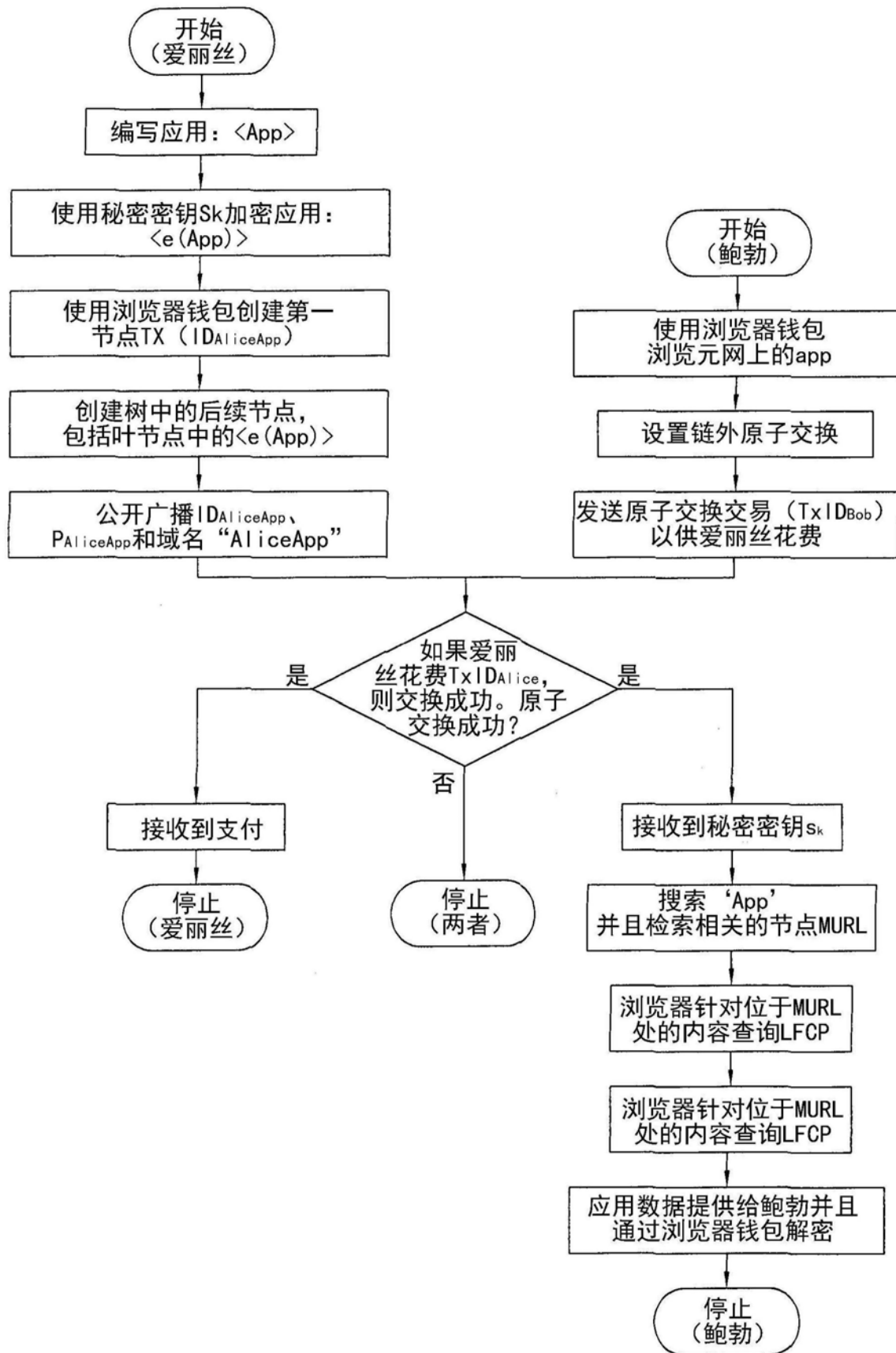


图19

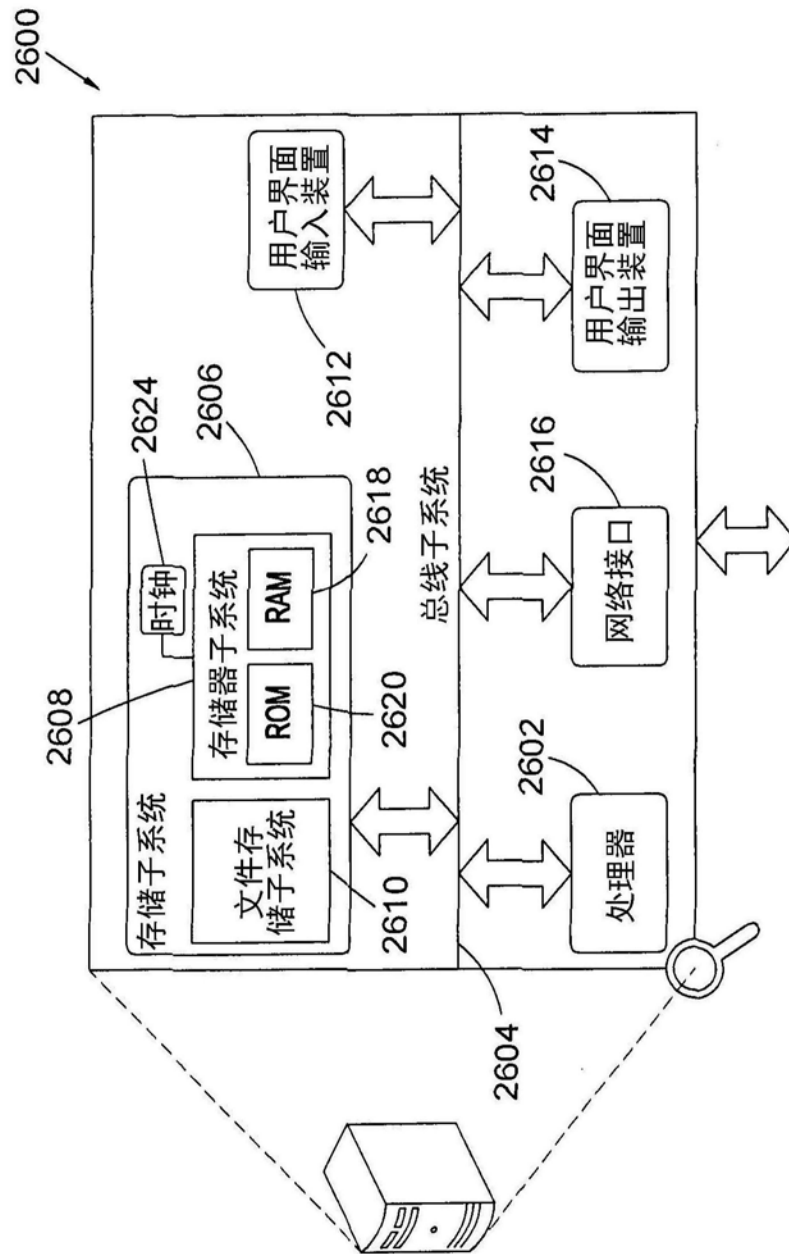


图20