一种基于联盟管理的高效分布式域名系统

邓锦禧[1], 韩 毅[1], 苏 申[1], 郭泽宇[1], 李 爽[1], 田志宏[1]

[1]广州大学 广州 中国 510006

摘要 在域名解析系统中, 下级域名的命脉被上级域名所掌握, 这种中心化的管理为域名解析带来了巨大的风险。以比特币等加密货币为代表的区块链则具有去中心化的特性。随着 namecoin 的提出, 区块链开始被应用在命名系统和域名解析的领域, 之后的 Blockstack 和 ENS 都提出了去中心化命名系统的解决方案。其中, Namecoin 和 Blockstack 采用了完全去中心化的命名管理方式, 产生了域名抢占问题。因此, 我们将目光转向了采用小群组投票决定域名增加和删除的联盟化管理方案。在联盟化管理方案中, 比如 ENS 的和超级账本的均存在交易空间太大的问题, 在区块链本身存储代价大的背景下, 存储效率将变得低下。因此, DNS 系统和区块链的结合难度很大, 不仅需要保证在多变的域名存储信息中保证存储总量较小, 同时还需要针对域名解析实现高效的联盟化管理, 这使得至今仍未有一个令人满意的去中心化域名解析系统的解决方案。为此, 我们提出了 ECMDNS——一个高效的基于联盟化管理的域名解析系统, 既考虑到 DNS 区域文件具有存储量大且变换频繁的特点, 又能在完全中心化和完全去中心化之间采取折衷方案, 并拥有较高的时空效率及较小的存储总量。我们通过区分链上链下的存储保证在多变的域名信息中保证存储总量较小; 通过群组决策投票的方式实现联盟化管理, 同时优化了 Hyperledger Fabric 提出的混合复制模型, 将空间存储效率优化到原本的 1/16。并且仅需要花费 1 次分布式副本同步, 就可以完成一项由 *n* 名成员背书对同一域名背书的事务, 并在联盟化管理的基础上实现区块链交易空间性能的高效性, 从而实现整体存储效率的高效性。

关键词 区块链; 域名解析系统; 联盟化管理

中图法分类号 TP311 DOI 号 10.19363/J.cnki.cn10-1380/tn.2024.01.07

An Efficient Decentralized Domain Name System Based on Consortium Management

DENG Jinxi[1], HAN Yi[1], SU Shen[1], GUO Zeyu[1], LI Shuang[1], TIAN Zhihong[1]

[1] Guangzhou University, Guangzhou 510006, China

Abstract In the domain name resolution system, the lifeblood of the subordinate domain name is controlled by the superior domain name. This centralized management brings great risks to domain name resolution. Blockchain, represented by cryptocurrencies such as Bitcoin, is decentralized. With the proposal of Namecoin, blockchain began to be applied in the field of naming system and domain name resolution, and then Blockstack and ENS proposed solutions of decentralized naming system. Among them, Namecoin and Blockstack adopted a completely decentralized naming management method, which caused the problem of domain name being squatted. Therefore, we turned our attention to consortium management that uses small group voting to decide the addition and deletion of domain names. Consortium management method such as ENS and Hyperledger Fabric, there is a problem that the transaction storage space is too large. Under the background of the high storage cost of the blockchain itself, the storage efficiency will become low. Thus, in practical application, the combination of DNS system and blockchain is very difficult. First, it is necessary to ensure a small amount of storage in the changeable domain name storage information. The second is the need to achieve efficient consortium management for domain name resolution, which makes there is still no satisfactory solution for a decentralized domain name resolution system. Therefore, we propose ECMDNS, an efficient domain name resolution system based on consortium management, which not only takes into account the characteristics of large storage and frequent transformation of DNS zone files, but also can take a compromise between complete centralization and complete decentralization, with high spatio-temporal efficiency and small storage volume. We keep the amount of storage small in the changing domain name information by differentiating the on-chain storage; in addition, the hybrid replication model proposed by Hyperledger Fabric is optimized to optimize the storage efficiency of space to 1/16 of the original. And it only takes 1 distributed copy synchronization to complete a transaction in which N members endorse the same domain name. And improve blockchain transaction space performance for consortium management, so as to optimize the efficiency of overall storage efficiency.

Key words blockchain; domain name resolution system; consortium management

通讯作者: 田志宏, 博士, 教授, Email: tianzhihong@gzhu.edu.cn。

本课题得到国家重点研发计划项目(No. 2018YFB1800701)资助。

收稿日期: 2022-05-06; 修改日期: 2022-06-27; 定稿日期: 2023-09-27

1 简介

域名解析系统本质上是一个分布式命名系统, 域名解析的目的是让 IP 地址更容易被寻址, 用户能通过一个容易记忆的域名找到其复杂的域名信息[1]。这种分布式命名系统的内容过多且难以管理, 因此现有的 DNS 采用了树形的中心化管理的方式, 上级域名对下级域名有完全的管理权力。但是这种中心化管理的方式会带来风险[2], 假如根域名的管理者删除某个国家根域名(ccTLD), 或者.com 域名的管理者删除某国家相关的跨国企业域名, 那么被删除的域名体系都将土崩瓦解。

随着比特币这种基于 P2P 网络和密码学算法的分布式存储系统的提出, 区块链的概念被引入[3]。接着, Namecoin 的提出说明了去中心化不仅可以应用在加密货币上, 也可以用于更广泛的命名系统[4]。由于 DNS 系统本质上也是一个以域名为主体的命名系统, 因此区块链能用于解决域名解析的中心化问题。

但是, 将 DNS 系统和区块链结合起来是非常困难的。Namecoin 曾经尝试用它的区块链存储域名信息, 并创立了.bit 这个去中心化域名。但是, 在处理区块链和域名信息的兼容性上, 它不能满足现今 DNS 系统的需求。它的域名是以先到先得的方式授权的[5], 并且每个名称都把其对应的全部内容存储在区块链上[6]。这种管理方式不仅面临着存储总量过大的问题, 更重要的是, 这是一种完全去中心化的管理方式, 先到先得的管理方式完全缺乏监管, 已经导致了严重的域名抢占问题。

由于域名解析系统的本质是命名系统, 我们也研究了一些较有名气的去中心化命名系统, 如 Blockstack 和 ENS。Blockstack 提出的多级验证结构在保证存储内容真实性的条件下大幅度缩小了去中心化命名系统的存储总量, 因为它只存储某个名字对应的数据哈希值或公钥代表, 而真实数据存储在下一级系统[7]。然而, Blockstack 在解决域名授权时, 仍采用了完全去中心化的管理方式, 并没有改善缺乏监管的问题。

在实践中, 中心化和完全去中心化的方式均表现出了它们存在的管理风险。因此在设计我们的系统时考虑到了一种基于联盟化管理的方式。中心化管理是一种由一个独立实体管理系统的管理方式, 去中心化管理是由系统所有参与者或者所有参与的独立实体共同管理系统的管理方式。我们考虑的联盟化管理是一种由部分的、有限个的独立实体管理系统。其中, ENS 在处理根域名时就用到了联盟化管理的方式, 它实现了一种基于多重签名智能合约的管理方式[8-9], 被授权的管理者可通过调用多重签名智能合约对根域名投票, 如果票数超过阈值, 则该域名可以被建立或删除。

然而 ENS 采用联盟化管理时在时间和空间上存在效率低下的问题。假如存在一个需要经过 n 名成员背书的域名, 则需要这 n 名成员分别发布调用这份多重签名智能合约的交易, 至少需要将 n 份交易以 PoW 共识的方式同步到所有的分布式节点上。因此, 实现这样的投票背书, 需要花费 n 次分布式副本同步, 以及在每个分布式节点上存储 n 份交易。然而, DNS 区域文件具有存储量大, 且变换频繁的特点(例如.com 域名拥有 1.47 亿个域名, 并且区域文件至少每天更改一次[10-11])。所以, ENS 投票决策带来的时空效率低下的问题, 在 DNS 系统中是无法接受的。

我们在深入研究 DNS 和区块链结合的难题后, 将需要解决的问题归结为两个。一是需要保证在多变的域名存储信息中保证存储总量较小。二是需要实现高效的联盟化管理方式。

为了解决以上问题, 我们提出了一种基于联盟管理的高效分布式域名系统 ECMDNS(Efficient Consortium Management DNS)。本文不仅通过构建联盟化管理的 DNS 系统, 还通过使用 MuSig 多重签名管理, 优化混合复制模型的投票机制。

为了保证在多变的域名信息中保证存储总量较小, 我们借鉴了 Blockstack 中的多级存储结构, 实现了区分链上链下的存储。在区块链节点上只保存密钥信息和完整区域文件的寻址信息, 完整的区域文件则存储在可自定义的存储位置, 并用区块链节点上的密钥签名验证其正确性。

为了保证在在去中心化的系统中保持一定程度的可监管性, 又要保证其时空效率。我们调查了 Hyperledger Fabric 的混合共识模型, 它把共识分为两个阶段: 用于生成背书信息的预共识阶段, 以及用于以分布式副本形式复制背书结果到整个分布式系统的 post 共识阶段。为了实现在投票效率上的优化, 我们重点优化了预共识阶段。我们将 MuSig 多重签名算法应用于投票信息背书上, 相较于 Hyperledger Fabric 原本的解决方案而言, ECMDNS 将空间存储效率优化到原本的 1/16。在时间效率上, 我们仅需要花费 1 次分布式副本同步, 就可以完成一项由 n 名成员对同一域名决策进行背书的事务。

本文系统可部署在任意级别的域名中, 可替代根域名、顶级域名等。由于现有中心化域名解析系统已经达到足够广泛的应用, 在具体实施时可通过

逐步替代的手段。在完全替代现有系统前, 但又未发生中心化风险时, 本文系统与现有域名解析系统并存, 并以现有系统数据为准, 原有域名解析系统数据假定为本文系统数据的超集。本文系统的入口为任意用户可自行搭建的 DNS 服务器, 访问这些 DNS 服务器时, 服务器将查询本系统的 DNS 数据并返回给用户。

如果在发展过程中发生中心化风险, 本文系统的联盟成员可选择停止与原 DNS 服务器的数据同步。此时本文数据将与原服务器数据不同, 并可通过联盟化管理的方式抵御中心化风险。我们的最终目标是替代原 DNS 服务器, 并建立由联盟化管理的 DNS 存储体系。

本文系统与原域名解析系统的区别只在数据来源, 因此并不影响用户域名解析的效率。

本文的主要贡献如下:

(1) 分析了当前 DNS 系统和区块链结合的现状, 并总结了两者结合过程产生的难题。

(2) 实现了一种符合当前 DNS 数据存储量大、变换频繁的特点的去中心化的存储方法。

(3) 优化了 Hyperledger Fabric 的混合共识模型, 实现了一种基于 MuSig 多重签名算法的、高效的、可用于 DNS 系统与区块链结合的联盟化管理方式。

(4) 提供 DNS 区域文件的存储和解析。ECMDNS 是一个能与现有 DNS 对接的、兼容性高的新型域名解析系统。

2 相关工作

我们深入调查了现有的基于区块链的域名解析系统、现有的共识系统、现有的背书策略以及现有的用户信息存储模型。我们从现有的基于区块链的域名解析系统中, 总结出了它们存在的存储冗余问题和监管性问题, 并从用户信息存储模型、共识系统、背书策略的角度深入调查和优化, 从而实现了 ECMDNS——一个能解决 DNS 与区块链结合而带来的存储冗余问题和监管性问题的去中心化域名解析系统。

2.1 现有的基于区块链的域名解析系统

Namecoin

Namecoin 是在 2011 年被提出的第一个基于区块链的命名系统, 并在其命名系统实现的基础上实现了域名解析。这是第一个达成有意义、唯一性(或称安全性)、去中心化三个条件的命名系统, 并实现了实时查询区块链的域名解析[4]。

Namecoin 采用了“先预定、再注册”的完全去中心化的方式处理名称相关的交易, 以避免在交易发起时发生域名抢占行为。但是, Namecoin 的注册方式在本质上依然是先来先服务的抢占注册, 而这还是导致了十分严重的域名抢占行为[5]。另外, Namecoin 将域名相关的全部信息以 json 形式存储在区块链上[6], 而区块链上的信息会以分布式副本的形式存储到所有的全量节点上, 这会大幅度提高存储总量。

Namecoin 是去中心化域名解析系统的先驱, 同时也给我们提出了一些关于如何实现一个去中心化域名解析系统应当考虑的问题。

Blockstack

Blockstack 是一个利用区块链来增强去中心化性的命名系统。它实现了多级验证的机制, 可利用现有区块链的强大算力来缓解区块链初创时期算力不足的问题。与 Namecoin 不同, Blockstack 没有把所有内容都存放在区块链上, 也没有成立自己的公链。它把数据分为多级存储, 每一级都会存储下一级的数据验证(比如公钥或哈希值), 它的最底层是以强大算力为保证的区块链[7,12-14]。

这样的多层验证机制, 既能保证用户可访问的资料的真实性, 又能大幅度缩减所需要的存储空间。但是, Blockstack 的域名注册方式与 Namecoin 相似, 它没有改变先到先得的注册制度[15], 仍是完全去中心化的管理机制, 所以还是存在十分严重的域名抢占问题。

Ethereum Naming Service (ENS)

ENS 是一套基于以太坊智能合约的命名系统, 它的目标是将人容易读懂的域名与人难以读懂或难以记忆的标识进行映射。比如, 它能实现 DNS 区域文件解析、以太坊地址解析和以太坊合约 ABI 解析等。

用户可以在 ENS 上设置一套与现有 DNS 相似的树形管理体系, 即下级域名需要在上级域名注册, 同时下级域名也会受到上级域名的管理。但是, ENS 的上级域名对下级域名的管理可以使用智能合约, 从而达到一种不能人为干涉的去中心化管理。常见的管理方式有先到先得管理和多重签名智能合约管理等[16]。

其中, ENS 根采用了一种联盟化的管理方式, 由多重签名智能合约投票来管理其唯一的顶级域名, 而投票者密钥则由以太坊社区可信任的若干成员掌握。这种管理方式在 ENS 这种唯一顶级域名的情况下显得非常有用, 因为它将去中心化的范围缩小到了可监管的小群体, 避免了先来先服务制度所带来的域名抢占问题。但是, 这种管理方式不适用于管理较多域名的情况。因为每次投票都需要发送一份交易, 使得它额外产生了更多次数的交易共识, 同时

也需要巨大的存储开销。

针对现有系统的分析

Namecoin 给出了一种用区块链存储真实域名信息的尝试, 而 Blockstack 和 ENS 则分别给出了一种基于区块链的命名系统的实现方式。从它们的工作中, 我们知晓了基于区块链的域名解析系统的可行性, 同时也知晓了将 DNS 与区块链结合的难度大, 其难度主要表现在以下几个方面:

需要实现高效的联盟化管理方式。Namecoin 和 Blockstack 均采用了完全去中心化的方式实现命名系统的管理, 使得域名的申请在完全未经验证的情况下完成, 造成域名完全缺乏监管, 导致了严重的域名抢占行为。ENS 通过多重签名智能合约实现一种联盟化管理, 从而实现在去中心化的情况下保持一定程度的可监管性。但是, 如果 ENS 需要完成一份带有 N 个投票对象的交易, 需要经过 N 次 PoW 共识和 N 次交易的提交, 它们分别带来了时间效率低和空间效率低的问题。

需要在多变的域名存储信息中保证存储总量较小。Namecoin 采用了将区域文件全量存储在区块链上的形式, 这样做有很好的去中心化的效果, 因为用户能在任意一个 Namecoin 节点上访问到完整的区域文件。但是, 区块链是一个以分布式副本为基础的存储系统, 所有的区域文件数据及其变更信息都将存储在所有的区块链节点上。DNS 区域文件不仅总量大, 而且变更频繁。例如.com 域名拥有 1.47 亿个域名, 并且区域文件至少每天更改一次[10-11]。如果我们按照 Namecoin 所用到的存储方式来存储 DNS 区域文件, 那么会造成非常大的存储冗余。

以上两个问题致使 DNS 系统和区块链结合的难度大, 使得很长的时间都没有能得到广泛应用的解决方案。为了解决现有系统的问题, 我们深入研究了现有的区块链共识系统、现有的去中心化的用户信息存储模型和用于投票的背书策略存储方法。

2.2 区块链和共识系统

区块链安全的本质在于将经过交易主体背书的交易, 通过共识系统以分布式副本的形式复制到节点不能互相信任的分布式系统中, 再借由庞大的分布式系统中大量的交易副本形成更强大的背书[15]。

以比特币和以太坊为代表的多数区块链公链的链上交易仅拥有少数交易主体, 每份交易均代表若干主体向若干客体转账, 或者某个主体调用智能合约的过程。因此它们仅需要少量的交易主体的签名就可以完成该交易的背书[3,17-18]。

Hyperledger Fabric 的背书过程比较复杂, 它提出的混合复制模型(hybrid replication)把共识分为两个阶段: 用于生成背书信息的预共识阶段, 以及用于以分布式副本形式复制背书结果到整个分布式系统的 post 共识阶段[19-20]。

在预共识阶段, 为了实现多种数字签名算法, Hyperledger Fabric 需要把所有投票者的数字签名记录在交易中。但这样的背书方式和存储方式产生了不少存储冗余。

在 post 共识阶段, 根据对提交交易成员身份的限制, 将区块链区分成公链和联盟链, 其中公链不设成员限制, 联盟链只允许经过验证的成员提交交易[21]。由于可提交交易的成员数量不同, 区块链需要采用不同的共识算法实现拜占庭容错, 联盟链共识算法的执行效率也大幅度优于公链的共识算法[22-23]。

关于时间性能, 在同等情况下, 预共识的时间性能花费很明显比 post 共识阶段的少。因为预共识只需要与交易相关主体进行交互, 而 post 共识阶段需要与包括交易相关主体在内的所有节点交互。

2.3 现有的用户信息存储模型

主流的用户信息模型和交易模型分为两种, UTXO-base 和 account-base, 分别以比特币和以太坊为各自代表[24]。UTXO-base 是以密钥为存储主体的多输入多输出的交易存储结构和用户信息模型, 并且每笔交易的输出不能多次使用。Account-base 模型是一种以存储内容的关键信息为主体、完整存储内容为客体的交易存储结构和用户信息模型, 存储内容的关键信息的类型可以随应用场景变换而变换(如: 在以太坊里指一串 160 位二进制数[18]、IPFS 里指数据哈希值[25]、域名解析存储里指域名)。

在基于区块链的域名解析系统中, Namecoin 的用户信息存储模型基于 UTXO-base 实现, 它的域名信息记录在其交易输出中, 存储时以密钥为主体, 域名及域名信息为密钥的对应客体。由于域名解析系统也是以域名为主体、域名信息为客体的模型, 所以 Namecoin 和 DNS 的存储主体不一样。因此 Namecoin 的交易中每一个域名所能存储的信息种类十分有限, 比如在一个域名对应多个密钥的需求中, UTXO-base 的存储模型将很难实现 DNS 信息的存储。

ENS 和 Blockstack 的交易信息基于 account-base 模型存储。这种用户信息存储模型的最大好处是能够改变存储内容关键信息的定义, 从而变动存储主体, 可以实现某个主体对应更多种类的信息存储。如果我们用 account-base 存储模型, 并且把关键信息定义为域名, 存储内容为具体的域名区域文件数据, 这样和现有的 DNS 系统的存储模型是相一致的。因

此, 在我们的系统中, 也采用了 account-base 的存储模型。

2.4 现有用于投票的背书策略的存储方法

投票是多个主体对同一客体背书的过程, 我们深入研究了 ENS 和 Hyperledger Fabric 用于投票的背书策略的生成和存储的过程。

ENS 根采用了由多重签名智能合约投票来管理顶级域名, 具体执行方式是各管理者各自向管理顶级域名的智能合约发布交易, 每份交易都需要经过以太坊的共识算法以分布式副本的方式复制到所有节点[8]。

Hyperledger Fabric 基于混合复制模型提出了一种用于投票的背书策略的存储方法, 以实现联盟化管理。在发布某份带有投票记录的交易前, 先收集所有投票主体对该交易的签名, 最后由发布者将这些签名记录在交易中, 再将交易以分布式副本的形式复制到所有节点[20]。Hyperledger Fabric 提出的存储方法解决了 ENS 共识次数过多的问题, 不管有多少个投票主体, 都只需要发布一份交易。

为了进一步解决背书策略带来的共识次数过多和签名数量过多的问题, 我们基于 Hyperledger Fabric 的混合共识模型, 研究了一套基于 MuSig 多重签名算法的、用于投票的背书策略的存储方法。

3 动机

前文提到了实现 DNS 系统与区块链结合的过程中, 存在两个基本问题:

(1) 需要在多变的域名存储信息中保证存储总量较小

为了解决存储总量问题, 我们实现了与Blockstack多级验证结构相似的区分链上链下的存储结构, 只将域名对应的密钥存储在链上, 而将完整的区域文件和与链上密钥对应的签名存储在链下。这样的存储方法能在多变的域名存储信息中保持存储总量较小, 因为当域名区域文件发生变动时, 我们并不需要在区块链上提交新的交易, 可以直接变动该区域文件及签名依然能保持完整的验证链, 即由区块链保证链上密钥的正确性, 由密钥和链下签名保证区域文件的完整性。区分链上链下的存储结构, 将域名具体规模与区块链系统区分开, 域名区域文件的大小只影响对应单个节点的域名服务器的效率, 而与区块链网络整体无关。

在设计区分链上链下的存储结构时, 我们设计了以域名为主体、域名信息为客体的基于 account-base 设计的用户信息存储模型。我们考虑到域名的数量是十分有限的, 因此在思考post共识阶段时, 设计了可基于现有的联盟链共识算法(如 RAFT 和 PBFT) [12,26], 实现新消息在分布式副本中同步。

(2) 需要实现高效的联盟化管理方式

为了解决联盟化管理的效率问题, 我们参考了 Hyperledger Fabric 的共识系统, 将共识划分为预共识和 post 共识两个阶段[20]。其中, 预共识是生成联盟背书策略的阶段(也可以说是交易生成的阶段), post 共识是将新消息复制到其他节点并验证的阶段。

我们使用了基于 schnorr scheme 的签名空间, 将 schnorr 签名算法[27]用作单重签名算法, 用于单人决策交易; 将 MuSig 签名算法[28]用作多重签名算法, 用于多人投票决策交易。我们通过这种方式优化了签名所花费的空间, 在投票交易中, 我们只需要记录所有投票者的索引编号。实验分析结果(section 7.2)表明在签名者较多时, 我们的方式所使用的存储空间约为 Hyperledger Fabric 的 1/16。

与 ENS 实现投票监管的方式相比, 我们不仅仅优化了空间存储效率, 同时也优化了时间性能。在 ENS 的多重签名智能合约中, 完成一次带有 N 名投票成员的域名申请需要提交 N 次交易才能完成共识[9]。我们实现的多人投票背书策略可在一份交易中保存这 N 次投票得到的背书结果, 因此只需要提交 1 次交易即可完成共识。

我们基于以上措施实现了 ECMDNS, 它分为命名系统和域名解析两部分内容。我们先实现一个去中心化的命名系统, 然后再在命名系统的基础上实现 DNS 资源记录存储, 最后由外部 DNS 服务器去读命名系统的域名信息, 从而实现域名解析。

命名系统基于区块链实现, 一共存在四种交易: 域名新建、域名弹劾、密钥更新、区域文件索引更新, 分别实现相应功能。其中, 域名新建和域名弹劾的交易利用混合复制模型结合 MuSig 多重签名的方式来实现多人背书, 密钥更新和区域文件索引更新这种仅需要单人背书的交易则利用 Schnorr 签名实现背书。

域名解析部分则是在已实现签名的命名系统中, 存储域名的相关公钥, 然后在完整的区域文件上附上其对应的签名用作验证。接着, 用离线备份的方式把区域文件存储在域名解析服务器, 并向用户提供域名解析。

4 命名系统

我们参考了 Blockstack 的多级存储方式, 实现了区分链上链下的存储。我们在区块链的链上存储了

用于验证交易的链上密钥、用于验证链下 DNS 区域文件数字签名的链下密钥以及域名区域文件存储索引。接着将 DNS 区域文件、根据链下密钥产生的 DNS 区域文件数字签名存储在存储索引对应的地方[7]。

为了实现影响域名增删的功能, 我们采用了更合理的投票方式, 而不是 Namecoin 和 Blockstack 的先来先服务的方式[5,15]。我们的命名系统参考了 account-base 的用户模型[24], 将每一个 account 定义为域名。它是一个以域名为主体的存储系统中。我们可以赋予某些域名参与投票, 并通过投票决定增加或者删除其他域名的权力。

虽然我们与 ENS 一样应用了多重签名算法实现多人投票交易的背书, 但我们使用先收集用户投票、再发布投票结果, 而非 ENS 投票者各自发布投票的方法。具体来讲, 就是基于优化的混合共识模型, 设计了以多人投票作为预共识阶段的背书策略, 以此来减少 post 共识的次数并且降低信息的存储总量。

4.1 用户信息存储

4.1.1 身份验证

我们采用非对称密钥的方式来实现身份验证的功能。其中, 密钥空间使用了与比特币和以太坊相同的 secp256k1 椭圆曲线[18,29], 并运用 schnorr signature scheme 为运算法则[27], 在具体实现上引用了一个 golang 实现的比特币全量节点库[30]。我们采用的包括+和*等运算都是基于 schnorr signature scheme 的运算。

设 secp256k1 生成元为 G, 私钥 $x \in (0, 2^{256})$, 则其对应公钥为 $x*G$, 这种运算的不可逆性由离散对数难题保证[31-32]。在存储密钥时, 我们将私钥分为 32 字节进行存储。对于公钥, 我们则先将 $x*G$ 分为点 (X_x, X_y), 再使用 64 字节来存储该点。当密钥被划分为字节来存储后, 可再用 hex 格式编码以用于传输。

对于某个密钥来说, 其公钥能被所有节点访问, 而私钥则只能被极少数的拥有者访问。

4.1.2 存储模型

在域名解析系统中, 域名的变更是不频繁的, 但域名的密钥变更是频繁的。所以我们也采用了 account-base 的这种以 account 作为存储的主体, 而非 UTXO-base 这种以密钥作为存储主体的用户信息存储模型。在设计信息存储方法时, 我们参考了 Blockstack 的多级验证结构[15], 将域名信息区分为链上信息和链下信息两类。其中链上信息包含了域名、链上密钥、链下密钥和域名区域文件索引。链下信息包含完整的域名区域文件和域名区域文件签名。

图 1 描述了 ECMDNS 的数据存储分布。其中, ECMDNS 的命名系统中存储链上数据, 完整的 DNS 数据存储在区块链以外。ECMDNS 实现的 account-base 模型中, account 指某个域名, 而内容指的是包含两种密钥和索引的链上数据。存储链上数据时, 将数据分为公开和私有两部分, 其中所有公钥和索引都属于公开信息, 它应能被所有存储节点访问, 而私钥属于私有信息, 只能被域名拥有者掌握的节点存储。

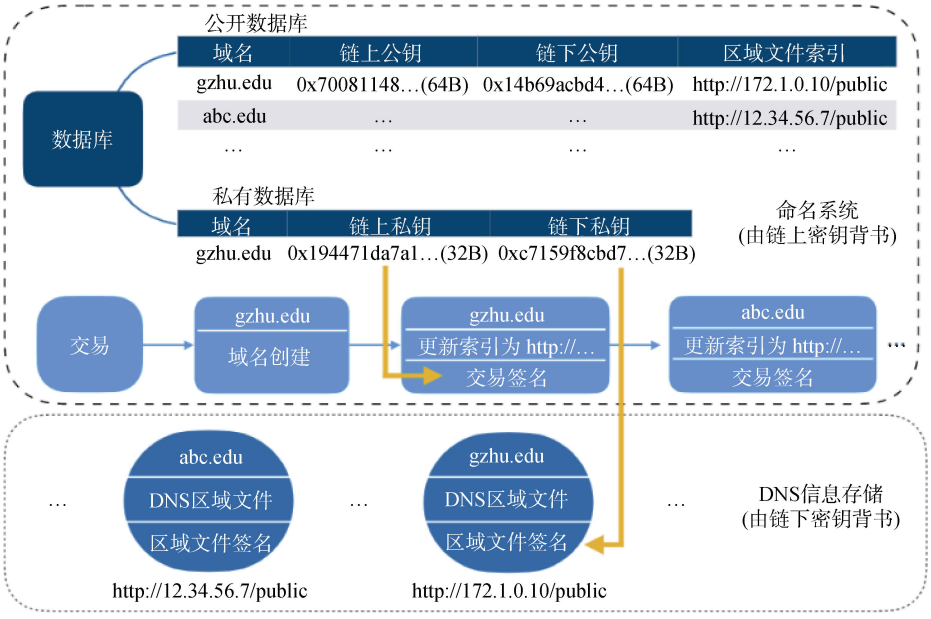


图 1 ECMDNS 数据存储分布图

Figure 1 ECMDNS data store distribution

其中链上命名系统为只存储 gzhu.edu 域名私钥的节点, 链下 DNS 数据存储系统可被任意用户访问

4.2 命名系统逻辑

Namecoin 先来先服务的注册方式, 以及抢占注册后的名字能永久被拥有者掌握的命名系统逻辑, 导致产生了很大的监督漏洞, 最终引发严重的域名抢占问题[5]。所以, 我们需要提出一种既符合去中心化的特性, 又能保证一定程度的可监管性的的基于联盟化管理的命名系统逻辑。

我们的逻辑总共包含四项: 新建域名、更新域名密钥、更新域名索引以及弹劾域名。其中, 每种逻辑在一次执行时都只需要一份交易。在 Namecoin 的基础上, 我们增加了删除域名的操作, 而域名的新建和删除的方式我们参考了 ENS 的做法——需要通过一个小群体的投票[8]。这种折衷的方式既考虑了域名去中心化的特性, 又考虑了一定程度的可监管性。拥有者更改自身信息的决策, 当且仅当拥有者自己同意才可以更改。这样做是为了保障拥有者自身的权益, 同时也能提高系统运行的效率。

对于更新域名密钥和更新域名索引的两项逻辑, 我们使用了 Schnorr 签名验证[27], 也就是在交易上添加该域名的链上数据密钥对应的签名。当其他节点接收到该类型的新交易时, 仅需要验证该交易的 Schnorr 签名是否与拥有者的对应即可。该部分将在 section 5.1 中详细描述。

对于新增域名和弹劾域名这两项影响域名总量的功能, 我们都应用了预共识的形式。先在线下收集所有投票成员的 MuSig 成员签名, 并聚合成多重签名记录在交易上, 从而完成预共识。当其他节点接收到通过 post 共识传输的交易后, 只需要提取出所有投票者的公钥, 再验证即可。该部分将在 section 5.2 中详细描述。

图 2 是命名系统的名称逻辑自动机, ECMDNS 的域名可以被群组投票创建和弹劾。域名在被创建和被弹劾的期间, 域名拥有者可以任意次更新对应域名的密钥和完整区域文件的索引。

4.3 交易内容

我们采用了 account-base 类型的交易, 每份交易的主体是某个域名, 其内容则是针对该域名的某种操作。系统的每份交易都包含了三项: 域名、交易项以及背书策略项。

其中, 交易项包含了四种交易的类型, 与前面提到的命名系统自动机一一对应, 也就是新增域名、弹劾域名、更新域名密钥和更新域名索引四项。

另外, 背书策略一共包含两种, 一种是需要单人签名的背书策略, 另一种则是需要群体决策的背书策略。其中, 新增域名和删除域名对应的是群体决策背书策略, 而更新域名密钥和更新域名索引则采用拥有者签名的背书策略。图 3 所对应的是交易背书策略和交易执行功能的对应关系。

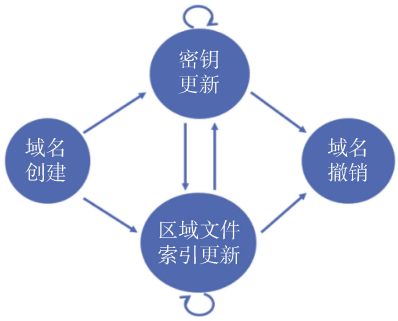


图 2 名称逻辑自动机

Figure 2 Name logical automata

域名可被群组决策创建和弹劾, 在创建和弹劾期间拥有者可以更新密钥和索引

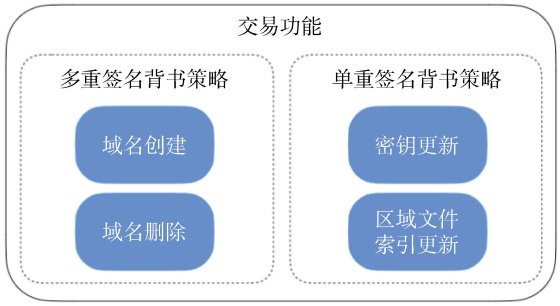


图 3 交易功能图

Figure 3 Transaction function diagram

每份交易在检验时, 验证者都需要先检验交易内容是否与背书策略一一对应, 然后再分别检验两项的正确性。

5 背书策略

ECMDNS 的两种背书策略都是基于 secp256k1 椭圆曲线和 schnorr signature scheme 执行。因此, 两种签名能使用同样的公私密钥对签发, 也就是用于签发单人 Schnorr 签名的密钥, 依然能用于签发 MuSig 成员签名并用于投票。只是生成单重签名仅需要单个密钥或单个签名者, 而生成多重签名交易需要多个密钥或多个签名者, 并且需要经过通信来交换密钥信息。两种背书策略的目的都是生成可以验证背书者或若干背书者的交易。

我们在执行签名时使用了 sha256 作为哈希函数[33-34], 用于将一段数据以不可逆的方式映射为 (0.2^{256}) 的正整数。我们用 $e=H(u)$ 表示将信息 u 通过哈希算法, 不可逆地映射为整数 e。其余的运算法则都是 schnorr signature scheme 的运算。在生成签名

时, 我们用到的信息包含了域名和交易项, 并用 m 表示这些信息。

在椭圆曲线上存在这样的性质: 对于每个相同的横坐标 x, 存在两个点与之对应, 将它们纵坐标记为 y_1 和 y_2, 存在 $y_1=-y_2$。我们使用雅可比(jacobian)投影压缩签名[33], 在二元集合 $S=\{y_1,y_2\}$ 内, 设 $y^*\in S$ 设:

$$\begin{cases} j(y^*)=1 \ if \ \exists n\in 2^{256}, n^2=y^* \\ j(y^*)=0 \ else \end{cases} \tag{1}$$

则有:

$$j(y_1)x \ or \quad j(y_2)=1 \tag{2}$$

5.1 单重签名背书策略

我们使用 Schnorr 签名来验证更新域名密钥和索引的交易。设域名拥有者的私钥为 x, 则拥有者公钥为:

$$(X_x, X_y)=X=x\cdot G \tag{3}$$

用户签名时生成随机整数 $r\in(0, 2^{256})$, 计算公开随机数:

$$(R_x, R_y)=R=r*G \tag{4}$$

如果 $j(R_y)=0$ 成立,则需要对 r 和 R 取相反数, 这样能保证 R_x 与 r 是一一对应的关系, 保存签名时只需要保存 R_x 即可节省空间。

在生成随机数 r, 并计算 R_x 完毕后。设每份交易中域名以及交易内容组成的文本为 m, 计算:

$$s=r+x*H(X,R_x,m) \tag{5}$$

并在交易上记录签名 (R_x,s)。

当其他节点接收到被广播的该类型交易时, 如果背书策略类型正确, 则先从数据库中提取出拥有者的公钥 P, 再根据交易中计算得到的 m 和签名 (R_x,s), 计算:

$$R'=s*G-P*H(P,R,m) \tag{6}$$

并验证:

$$R'_x=R_x \wedge j(R'_y)=1 \tag{7}$$

如果都成立则签名正确。

5.2 多重签名背书策略

ECMDNS 采取预共识过程实现多重签名背书策略, 而预共识过程通过 MuSig 多重签名算法[28]实现的。整个过程就是先收集若干投票者的 MuSig 成员签名, 再聚合为多重签名, 最后记录在交易上。我们不需要像 Hyperledger Fabric 的实现方式一样记录 n 个签名, 只需要记录 n 个投票者索引和 1 个 MuSig 多重签名即可, 这节省了交易的占用空间。

设需要 n 名用户对某份交易进行投票, 各自私钥为 x^i, 公钥 $X^i=x^i*G$。 投票需要经历三阶段, 下面凡涉及到请求者向投票者发起请求部分均可以并行执行, 不存在需要先接收到某个投票者的结果后才能接受另一个的情况。图 4 表示预共识过程申请多重签名的信息交互情况, 具体分为以下三个阶段:

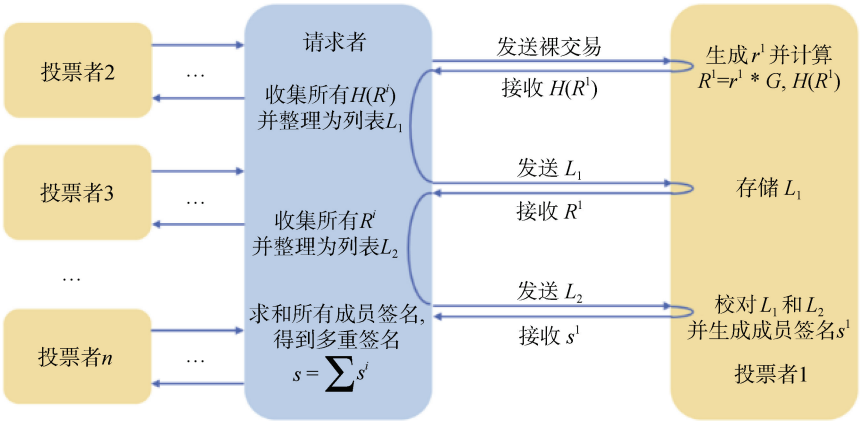


图 4 申请 MuSig 多重签名的三步交互过程

Figure 4 Three-step interactive process of applying for MuSig

随机数哈希交换: 请求者确定需要请求哪些投票者对这份交易投票, 将交易信息 m 发送至这些投票者。投票者接收到投票请求后, 如果选择向该交易投票, 则需要自行生成私有随机数 $r^i\in(0,2^{256})$, 并计算公开随机数 $R^i=r^i*G$ 及其哈希值 $H(R^i)$, 然后将这些数保存在各自的本地。投票者将 $H(R^i)$ 返回至请求者之后, 请求者就将所有投票者对应的 $H(R^i)$ 记录在本地, 并计算序列 L_1:

$$L_1=\langle H(R^1),H(R^2),H(R^3),\cdots\rangle \tag{8}$$

随机数交换: 如果请求者收到所有投票者的随机数哈希, 则代表他们都初步同意该交易。接下来, 请求者将 L_1 发送至所有投票者。在接收到 L_1 后, 投票者先检查列表里是否存在自己先前生成的公开随机数哈希。如果存在, 投票者则将 L_1 存储在本地, 并将 R_i 返回给请求者。之后, 请求者将所有投票者对应的 R_i 记录在本地, 并令列表 L_2:

$$L_2 = \left\langle R^1, R^2, R^3, \cdots \right\rangle \tag{9}$$

成员签名请求: 请求者先自行计算 L_1 和 L_2 是否一一对应, 即判断:

$$\forall i \in [0, n), L_1^i = H\left(L_2^i\right) \tag{10}$$

如果随机数检验结果匹配, 请求者则将 L_2 发送至所有投票者。

所有投票者再各自检验 L_1 和 L_2 是否匹配, 接着各自计算聚合随机数:

$$R = \sum R^i \tag{11}$$

与单重签名相似, 我们需要 $\sum r^i$ 和 R_x 一一对应, 因此需要先计算 $j(R_y) = 1$ 是否成立。如果不成立, 投票者则各自都将私钥随机数 r^i 以及聚合随机数 R 取反。因为 $R = \sum R^i = \sum r^i * G$, 若所有 r_i 都取相反数, 最终得到 $-R$ 满足 $R_x = (-R)_x, R_y = -(-R)_y$, 则有 $j((-R)_y) = 1$。

接着计算聚合公钥 $X = \sum X^i$ 得到, 令有序列表 $L_3 = \left\{P^1, P^2, P^3, \cdots\right\}$, 所有投票者都需要按照相同规则计算 L_3。最后成员签名则为:

$$s^i := r^i + H(X, R, H(m)) * x^i * H(L_3, X^i) \tag{12}$$

最后投票者各自向请求者返回成员签名 s^i。请求者得到所有成员签名后, 计算聚合签名:

$$\mathrm{s} := \sum s^i \tag{13}$$

并在交易上记录签名 (R_x, s)。

生成签名后, 请求者需要检验 (R_x, s) 是否正确, 其检验过程和单重签名相似。先根据所有投票者公钥 P^i, 计算聚合公钥 $P = \sum P^i$。并根据已有信息, 计算期望的聚合随机数:

$$R' := S * G - H(X, R_x, H(m)) * \sum (X^i * H(L_3, X^i)) \tag{14}$$

并验证:

$$R'_x = R_x \wedge j(R'_y) = 1 \tag{15}$$

是否成立, 如果成立则签名正确。

若检验签名正确, 请求者需要记录域名索引列表 L_4, 其中 L_4^i 是一个在ECMDNS中与该域名对应的整数型索引号。表示该域名是历史中第几个域名。

最后, 请求者将 (L_4, R_x, s) 记录在交易中, 完成预共识阶段。

交易生成完毕后, 即可执行 post 共识。其余节点经过 post 共识接收到新交易后, 需要先检验列表 L_4 是否符合投票规则, 如果符合条件则根据 L_4 读取出所需要公钥 X^i, 并检验 (R_x, s) 是否正确。

5.3 投票法则

投票法则指生成多重签名算法过程中, 投票者决定是否配合请求者完成多重签名背书策略的法则。在新增域名或删除域名的投票法则中, 本应当由用户自行设计投票法则。但考虑到现有的去中心化域名解析系统中, 还没有投票法则的设计, 因此我们给出一种实际应用中, 所使用的投票法则的设计方案。

现有的基于联盟化管理投票方案中, ENS 用多重签名智能合约的方式实现实时的去中心化的票数统计, 每个用户能随时调用多重签名智能合约更改自己的选票。本文所实现的联盟化管理的背书策略与Hyperledger Fabric都基于混合共识模型实现, 两者的投票都需要一次性获取所有投票者的选票。因此本文系统应在投票申请发起时, 投票者应该要在由网络时延决定的短时间内, 完成对某个域名的决策。

因此, 我们要求所有投票者均实现了能短时间做出决策的方案。假定了所有投票者都有自己的人工审核机制, 在生成多重签名背书策略前, 请求者需要将需要执行的交易发布至投票者完成人工审核。若人工审核通过, 投票者则在服务器中记录下该交易的哈希值, 并向请求者返回使用该投票者链上密钥签署的数字签名, 保证不可抵赖。在投票者接收到某个请求者发来的交易时, 查看数据库是否存在该交易的哈希值, 就能知道是否有通过线下人工审核, 并作出是否投票的判断。

该投票法则仅作为设计参考, 实际投票法则应该又投票者自定义。

5.4 交易的生成和执行

无论何种交易, 它的生成都经历过读取本地数据库、编写交易内容、根据交易类型编写背书策略并执行 post 共识。每个共识节点接收新交易时, 都需经过先检验背书策略、再写入数据库的过程。所有交易内容的生成所依赖的数据都来自链上的公开数据库, 而背书策略的生成会依赖非公开数据库里存放的私钥信息。

系统的交易生成流程如图 5 所示。在交易生成

时, 我们需要先确定所需要操作的域名和交易执行的具体操作, 接着根据交易执行的操作选择需要的背书策略。如果交易要执行的操作需要用到单重签名背书策略, 则读取该域名对应链上私钥写入 Schnorr 签名; 如果交易要执行的操作需要用到多重签名背书策略, 则请求者需要将交易发送至所有投票者来请求多重签名, 并依据投票者信息以及多重签名生成背书策略项, 将背书策略项写入交易中。然后发起 post 共识, 直到所有接收者更新自身状态。

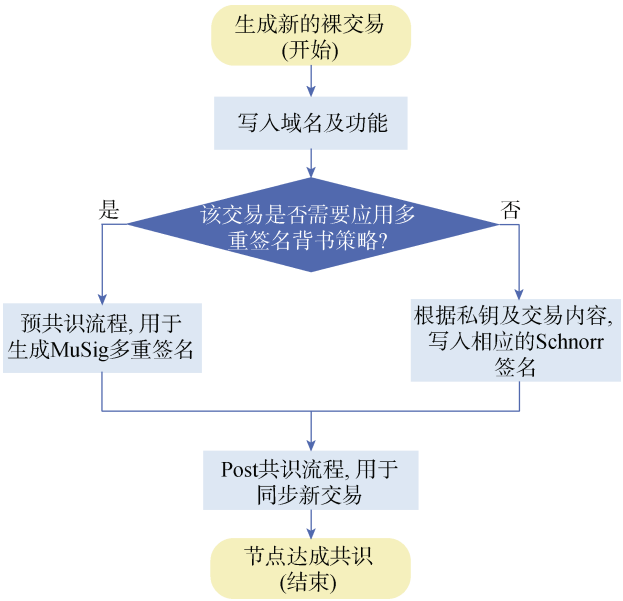


图 5 交易生成流程图

Figure 5 Transaction generation flowchart

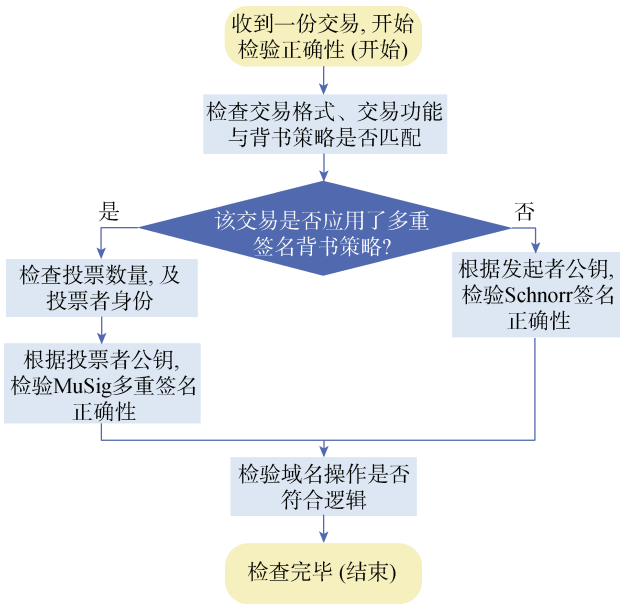


图 6 交易验证流程图

Figure 6 Transaction validation flowchart

系统的交易验证流程如图 6 所示。当某节点在 post 共识过程中接收到一份新交易时, 需要先检验背书策略类型和交易的操作类型是否匹配。如果交易使用了单重签名的背书策略, 则需要读取域名拥有者公钥, 检验交易上的 Schnorr 签名与域名拥有者是否匹配; 如果交易使用了多重签名的背书策略, 则先检验签名成员和身份是否符合要求, 若符合则读取所有签名者公钥, 检验 MuSig 多重签名与这些签名者的公钥是否匹配。若匹配, 仍需要检验交易的操作是否符合命名系统逻辑, 比如域名在创建时是否已存在该域名、域名删除时该域名是否存在。如果其中一项检查不通过, 则检查会中断, 该交易就不能通过检验。

6 DNS 信息存储

在执行一次 DNS 查询时, 用户一般是以标准的 DNS 请求访问某递归服务器, 然后递归服务器通过迭代查询按照树形解析的方式由根开始直到查询到目标的 IP 地址。ECMDNS 本质上是一个扁平化的查询系统, 它不存在现有系统的树形结构, 所以它可以实现去中心化的单层域名解析。ECMDNS 的命名系统以去中心化的方式存储了每个域名所对应的完整区域文件的索引以及其链下验证密钥。ECMDNS 的 DNS 存储操作根据操作者主要分为两种: 拥有者操作和解析者操作。

拥有者操作: 拥有者的定义是某个域名拥有其链下私钥的实体, 它需要在不改变区块链链上数据的情况下更改 DNS 区域文件。在命名系统中存储了该域名对应的索引, 并且该索引是公开的。拥有者需要将该域名对应的完整域文件, 以及其使用域名链下私钥生成的区域文件的 Schnorr 签名一同存放在该索引对应的位置, 并且该区域文件需要符合 DNS 区域文件的格式[34]。因此, 拥有者需要使用链下私钥生成该区域文件的 Schnorr 签名, 并将该签名和区域文件存储在其对应索引的位置, 其签名的方法与第 5.1 章提到的单重签名的生成及验证方法相同。

解析者操作: 解析者是任何能通过某域名索引, 能够访问并得到其完整区域文件和签名, 并能使用其链下公钥验证签名正确性的实体。解析者面向的是广大用户, 它需要以标准的 DNS 请求和答复的方式与用户交互[34]。ECMDNS 所采用的方式是服务器定时访问命名系统, 得到其关注的域名索引和公钥, 得到其对应的区域文件和签名, 最后使用链下公钥验证签名的正确性。如果知晓该区域文件是由拥有者发布的, 解析者则将区域文件离线复制至本地 DNS 服务器中, 使得外网用户可以用原有的方式访

问 DNS 服务。

Namecoin 使用的是将完整文件存储在链上的方式, 并开发了可实时根据接收到的 DNS 请求读取节点信息的 Namecoin-DNS[35]。我们认为这样的存储方式在域名区域文件较大时, 链上存储总量太大。由于区块链链上空间存储代价太大, ECMDNS 选择了区分链上链下的存储方式。

但这样的方式相比 Namecoin 而言不能保证区域文件的去中心化, 所以额外添加了区域文件签名用于验证拥有者。因此我们能在区块链存储总量较小的条件下完成分布式授权。同时, 我们认为 DNS 服务器的访问量会比区块链节点的访问量大, 如果使用实时访问的方式会增加区块链节点的负担, 因此采用了离线 DNS 区域文件的方式存储。

7 应用及分析

7.1 应用

我们使用了 raft 作为 post 共识部分的共识算法[12], 实现了整个完整的系统, 并部署在若干服务器中, 每台服务器都模拟一个 raft 共识节点。在我们实现的 raft 共识算法中, 每份交易都会经过由若干节点投票。若票数过半则更新当前状态, 投票原则是检验交易在当前数据库中的正确性。我们使用了 go 1.14 实现了每个共识节点的逻辑, 并将每个共识节点部署在一台服务器上, 服务器之间以星形拓扑连接, 单台服务器配置包括操作系统 Ubuntu 18.04、单核 CPU OctaCore Intel(R) Xeon(R) Silver 4116 CPU @ 2.10GHz、1G 内存、50G 硬盘。由于用到计算资源比较多, 因此我们把实验环境部署在鹏城实验室的靶场上。

实现 raft 算法时, 我们侧重考虑了两个重要方面, 领导者选举和新消息投票传递。领导者选举是我们实现主节点选举的过程, 它在初始化及原主节点失效时发生。我们在所有节点中内置了时钟, 当时钟归零则判定为超时。其中, 主节点心跳包、候选者请求等都可将时钟重置为随机的 3～6s, 而成功选举为候选节点可将时钟重置为 10～15s, 并且每 0.5s 检测一次是否超时。

当需要发起新消息时, 主节点每 0.5s 向全部节点广播最新的交易或者广播待更新交易的投票请求。如果收到一半以上节点投票, 则更新自身状态, 并在心跳包中将最新状态扩散出去。在实验中, 为了让尽可能多的节点都按序收到最新状态, 我们强迫主节点至少完成两次广播后才可以发起新一轮的新交易投票。

7.2 交易空间性能分析

空间性能分析包括两项, 一般性分析和实例分析。在一般性分析中, 我们主要研究系统在联盟成员增加时的可扩展性; 实例分析中, 我们主要研究系统在实际运行时, 所产生的具体开销是否合理。

7.2.1 一般性分析

在一般性分析中, 我们假设域名长度为 D, 私钥长度为 X, 公钥和签名长度为 $2X$, 域名完整区域文件索引长度为 Y, 单位为字节。每份交易中都包含域名、交易项、背书策略项三项, 其中交易项和背书策略项的对应关系已在 4.3 章提及。其中, 我们用 1 字节的空间表示交易类型和背书策略类型。

a)*单重签名交易性能*

使用单重签名背书策略的交易共包含两种类型, 更新域名密钥、更新域名索引。

更新域名密钥交易: 该交易的交易内容包含两部分, 更新后域名的链上公钥、更新后域名的链下公钥, 在交易执行后, 该域名的验证都按照新的密钥进行。其交易内容包括两个公钥和占用 1 字节空间的交易类型, 总共为 $4X+1$ 字节。背书策略项中, 单重签名大小为 $2X$ 字节, 因此加上背书策略类型的 1 字节后, 背书策略项总共 $2X+1$ 字节。最终的交易包含了域名、交易内容和背书策略, 总共大小为 $D+6X+2$。

更新域名索引交易: 该交易的交易内容只包含更新后的域名索引一项, 交易执行后, DNS 服务节点将从新的索引获取区域文件, 其交易内容大小为 $Y+1$。其背书策略大小同样为 $2X+1$, 因此总共大小为 $D+2X+Y+2$。

因此, 单重签名交易的大小仅与域名长度、私钥长度、域名区域文件索引长度有关。而在现有系统中, 这些项的长度都是有限的, 因此单重签名交易的空间性能可扩展性良好。

b)*多重签名交易性能*

使用多重签名背书策略的交易共包含新建域名和弹劾域名两种, 由于两种交易全部的信息都记录在域名以及交易类型种, 因此它们的交易项均只占用 1 字节。

该类型交易需要在预共识阶段需要收集投票成员的投票信息, 设需要生成一份带有 n 个投票成员的交易。在多重签名的背书策略内容中, 包含索引列表和签名两部分。其中索引列表包含索引数量和若干具体的索引值。其中, 每个索引与域名是一一对应关系, 每个索引表示生成的第几个域名, 如果一个在链上的域名先被删除再被新建, 它的索引值会被

改变。我们假设域名索引的长度为 B 字节, 则可存储的历史域名总量为 n_1=256^B, 同理 $B=\frac{1}{8}\times\log_2 n_1$。对于一份历史存储域名上限为 n_1, 有 n_2 名成员参与投票的交易, 其背书策略项占用空间包括了该交易参与投票的域名数量(该处假定为 4 字节)、n_2 个投票成员索引以及 MuSig 多重签名。背书策略项占用的总空间为 $4+B+2X$。再加上其域名和交易项总共大小为 $D+1$ 的空间, 其总空间大小为 5+D+2X+B。记交易空间为 S, 记 K=5+D+2X, 则:

$$S(n_1,n_2)=\frac{n_2}{8}\times\log_2 n_1+K \tag{16}$$

如果采用 Hyperledger Fabric 的做法, 把所有签名都记录在交易上, 相同条件下, 其交易大小为 S':

$$S'(n_2)=n_2*2X+K \tag{17}$$

$$S/S'=\frac{\log_2 n_1}{16X} \tag{18}$$

我们可以看出, 如果历史存储域名上限 n_1 越大, S 的大小就越接近 S'。当 $n_1<2^{16X}$ 时, $S<S'$ 均成立。

7.2.2 实例分析

以域名 gzhu.edu.cn 为例, 设其区域文件索引为“http://202.192.18.1/public”, 简单计算可知, 域名占用空间 D=11 字节, 域名索引占用空间 Y=26 字节。假设我们使用 X=32 字节的私钥, 其公钥和签名均为 64 字节。

在单重签名背书策略的交易中, 代入一般性分析的结果, 一份更新域名密钥交易总共占用 208 字节, 更新域名索引的交易总共占用 103 字节。显然, 该存储空间占用大小的交易在合理的范围内。

在多重签名背书策略的交易中, 当域名数量较多时签名所占用的空间远大于其他信息所占用空间, 因此 K 值大小可忽略不计。本文系统调查了.com 域名数量为 1.47 亿个[36], 因此在对比实验中设定域名上限分别为 2^{32} 和 2^{64} 两种情况进行比较, 在密钥长度相同的情况下与超级账本所实现的累加签名预共识的方法进行比较。以下分别对 256 位和 512 位两种密钥长度分别做了实验。

当设密钥长度为定值时, 从图 7 和图 8 中均显现相同的趋势, 累加签名预共识方法所生成的交易大小远远大于域名上限分别为 2^{32} 和 2^{64}。其中, 若将签名数量为现有的国家根数量(250 个), 在 256 为长度的密钥前提下, 原本的累加签名预共识交易大小为 15.6kB, 而上限为 2^{32} 个域名的 MuSig 预共识交易仅需要 1.04kB。如果考虑可扩展性, 即便将上限扩展为 2^{64} 个交易也仅占用 2.02kB 的空间, 仍远远小于累加签名预共识的 15.6kB。

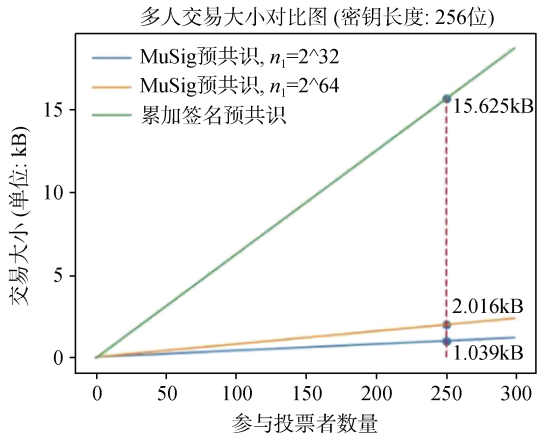


图 7 多重签名背书策略交易对比图(256 位密钥)

Figure 7 Multi-signature endorsement strategy transaction comparison diagram (256-bit key)

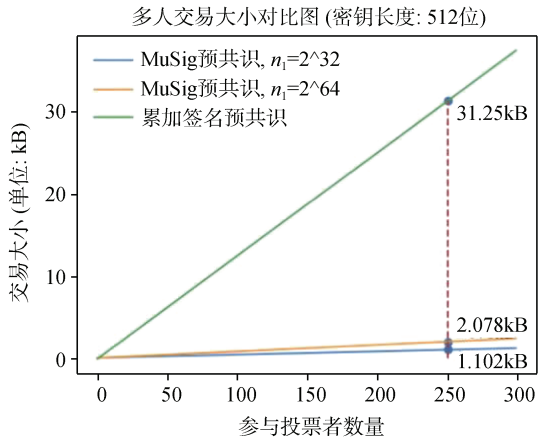


图 8 多重签名背书策略交易对比图(256 位密钥)

Figure 8 Multi-signature endorsement strategy transaction comparison diagram (256-bit key)

若采用更安全的方式, 采用 512 位长度的密钥, 在同样用 250 个签名者的情况下。累加签名预共识交易大小为 31.25kB, 而域名个数上限为 2^{32} 时占用仅需要 1.10kB, 考虑可扩展性设定上限为 2^{64} 也仅占用 2.08kB 空间。

在以上两组对比实验中能看出, 在密钥长度相同时, MuSig 预共识生成的交易比超级账本提出的混合共识模型预共识生成的交易, 在空间占用上有显著优势。并且如果以现有共 250 个拉丁字符格式(Latin-character) ccTLD 形成一份投票交易为例, 空间占用的优势将变得更加明显。

比特币从 2020 年 6 月至 2020 年末的平均交易大小为 471 字节[37], 而一份拥有 250 个国家根背书的交易约占用 2 份比特币大小的空间, 说明使用多重

签名背书策略的空间占用是合理的。而更新域名密钥交易约占用 0.43 份比特币平均交易大小的空间，而更新索引交易仅占用 0.21 份。

在 MuSig 预共识中，图 6 (a)和图 6 (b)中均存在相同趋势：当域名上限 n_1 增加时，空间都变得更大。因此，需要在密钥长度相同情况下对比可扩展性。根据公式(18)可绘制图 7。

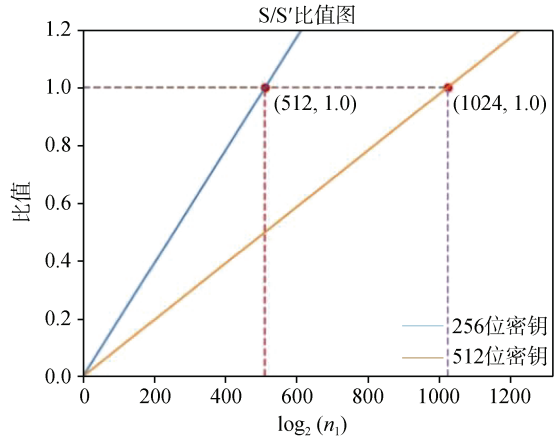


图 9　预共识交易空间比值图

Figure 9　Ratio diagram of pre-consensus trading space

由图 9 可看出，在 256 位密钥的前提下，在历史域名总数在 $n_1<2^{512}$ 时，MuSig 使用预共识的方案比原有混合模型方案要好；而在 512 位密钥前提下，在历史域名总数在 $n_1<2^{1024}$ 时，MuSig 使用预共识的方案比原有混合模型方案要好；而这个域名数量在现有可见的水平下，是不可能达到的。

综上所述，本文系统的交易在空间性能上是合理的，且具有良好的可扩展性。

7.3　时间性能分析

时间性能分析主要包括两项，第一项是针对我们系统优化的预共识阶段的时间性能分析，第二项是整个完整实现的系统的吞吐量分析。时间性能分析的目的在于检验系统的在预共识阶段付出额外时间代价，以及整个系统的吞吐量是否在合理。

7.3.1　预共识时间性能分析

本文测量了预共识阶段在整个共识阶段的时间占用情况。实验分为两组，第一组包含 5 个共识节点，第二组包含 10 个共识节点。其中第一组实验中，每个共识节点包含 20 个投票者密钥；第二组实验中，每个共识节点包含 10 个投票者密钥。两组实验中，网络都总共包含 100 个投票者。

每次实验中，我们都会生成一份由 $x\in[1,100]$ 名投票成员投票的创建某个随机域名的交易。分别记录每个 x 测量得到的预共识花费时间、传播完成时间、同步完成时间。三个时间测量指标都是从域名和交易项生成完成后开始计算，预共识完成实际、传播完成、同步完成是一个交易从生成到复制到所有节点所需要完成的三个阶段。预共识完成时间是指完成多重签名背书策略的生成所花费的时间，传播完成时间指交易被主节点接收所花费的时间，同步完成时间指交易被复制到所有节点上所花费的时间。

实验结果如图 10 所示，两组对比试验有相同趋势，只是第二组实验中传播完成时间和同步完成时间波动比较大，特别是同步完成时间这一项。产生波动的原因有两个：一是数据传播量会随着节点数量增多而增多，二是同步完成的条件会随着节点数量增多而变得苛刻，两者都会造成更大的波动。

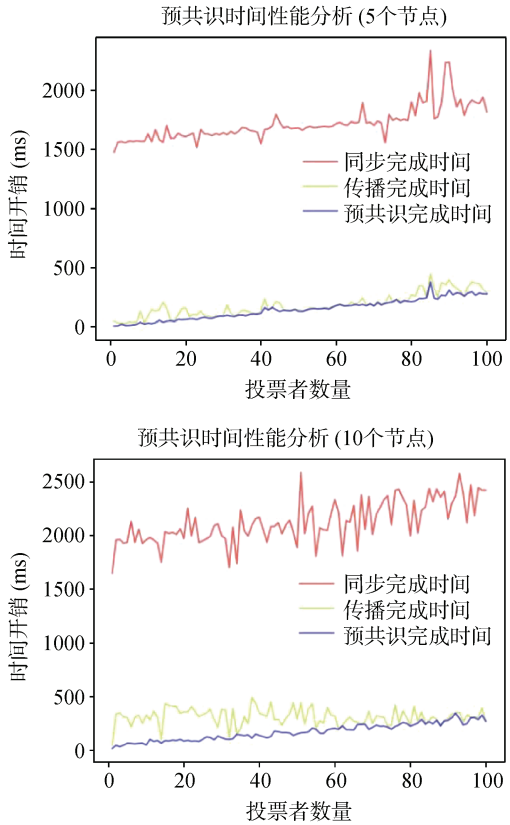


图 10　预共识时间性能分析

Figure 10　Pre-consensus time performance analysis

但即使产生了这样的波动，依然显示出相同的趋势。两组实验的预共识完成时间都随着域名数量的增加呈线性增加的趋势，而且即使在投票成员数量到达 100 的条件下，预共识时间在同步完成时间中占比依然较小。这是由于预共识过程中，只有第一次接收的时候需要读取当前数据库，之后的投票阶段都能很快的完成，运算也只是无状态的椭圆曲线运算。而 post 共识是有状态的，实验中的 raft 算法需

要等待每一项工作完成才能进入下一阶段的运算, 比如我们会强迫主节点需要广播两次当前状态才能发起下一次的投票, 这个过程是必要的也是耗时的。在我们的实验中, post 共识的时间体现在同步完成时间与发送完成时间之差, 实验结果也显示它们的差异比较大。

实验说明了我们成功地改进了 hyperledger fabric 的混合复制模型, 基于 MuSig 的预共识阶段产生的额外时间代价在可接受范围内。

7.3.2 吞吐量分析

我们测量了区块链网络的吞吐量, 得到若干交易所需要的同步时间, 从而知晓 ECMDNS 在共识上的时间优化。ENS 是通过多重签名智能合约实现了投票, 每次成员投票都需要发布一份交易完成, 也就是执行一次 post 共识。因此, 我们需要知道执行多次 post 共识所花费的时间, 与前面预共识时间对比就可以知道优化了多少。

我们向共识网络来连续发送了 100 份更新域名索引的交易, 模拟 ENS 向多重签名智能合约投票过程。每份交易都记录了它的传播完成时间和同步完成时间。由于我们采用的是异步发送, 我们设横坐标 x_i 的传播完成时间是第 x_i 份交易传播完成所花费的时间, 同样 x_i 的同步完成时间是第 x_i 份交易同步完成所花费的时间。与预共识性能分析相同, 同样设置两组对比测试, 5 节点网络和 10 节点网络。

实验结果如图 11 所示, 在两组测试中, 完成传播或同步的序号几乎与对应时间成线性关系, 而且相同的序号对应的传播完成时间和同步完成时间之差比较接近, 这个时间也与前面测得 post-consensus 时间相似。再次说明了预共识的时间性能分析中的即使同步时间波动较大, 也是合理的。

进一步, 如果我们采用 ENS 类似方案, 假若需要得到一份 n 名成员同意的决议, 需要发起并同步 n 份交易。我们的实验结果显示这样所耗费的时间成本是巨大的, 因为它需要 n 份交易达成共识。比如需要通过一项由 100 名成员的决议, 模拟 ENS 所花费的时间约为 180s, 而采用混合共识的方式只需要 2.5s 就可以完成。ECMDNS 采用混合共识模型优化共识次数是十分有效的。

8 结论和未来工作

我们提出了 ECMDNS, 一个具基于联盟化管理的、存储总量较小的、共识次数较少的域名解析系统。首先, 我们采用了区分链上链下存储结构, 解决了域名区域文件过大导致区块链链上节点存储总量过大的问题; 接着, 我们采用混合复制模型并做了优化, 在预共识阶段应用MuSig多重签名算法, 解决了投票共识次数过多的问题, 并进一步减少了链上节点存储总量。在前两者的基础上, 我们实现了基于联盟化管理的命名系统逻辑, 为 ECMDNS 加入了群组投票和删除的功能, 实现了在去中心化条件下的一定程度的可监管性。

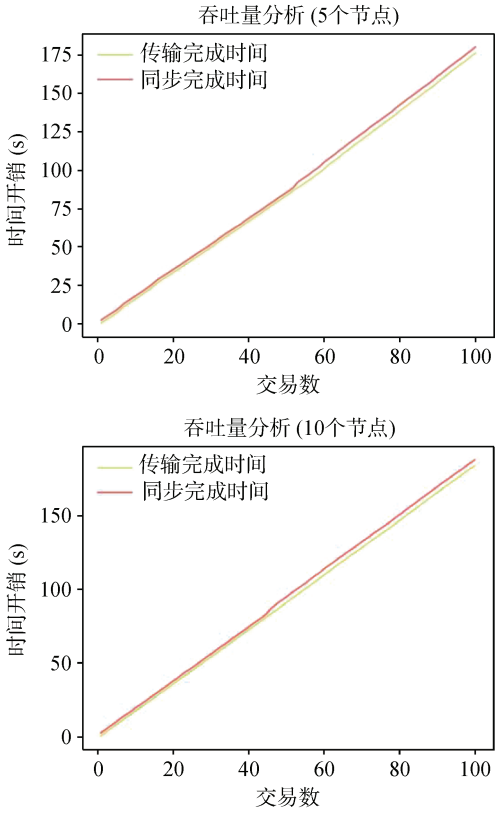


图 11 吞吐量分析

Figure 11 Throughput analysis

我们给出了 ECMDNS 的一种具体应用方法, 在可自定义的 post 共识阶段中采用 raft 算法(以及投票法则), 并分析了空间和时间性能。实验结果显示, 我们采用 MuSig 多重签名算法将原本混合共识模型的交易优化到原来的 1/16, 并且时间成本比较小。此外, 我们减少了投票交易的共识次数, 大幅度减少了决策完成的时间。

未来, 我们将进一步优化 ECMDNS。我们将继续针对去中心化 DNS 系统研究存储优化方案。另外, 我们也将进一步研究共识算法方面的问题。本文我们主要优化了混合共识模型中预共识过程, 在 post 共识过程甚至在混合共识模型本身仍有不少优化的空间。未来的 ECMDNS 将会进一步和现有 DNS 系统结合, 其实现会逐渐透明化。

致谢 国家重点研发计划项目(No. 2018YFB1800701); 广东省重点研发计划项目(No. 2020B0101090003); 国家自然科学基金项目(No. 61902083, No. 62172115, No. 61976064), 广东省高校创新团队(No. 2020KCXTD007)与广州市高校创新团队(No. 202032854), 以及广州市市校基础研究计划联合资助项目(No. 202102010445)。

参考文献

[1] Mockapetris P V, Dunlap K J. Development of the Domain Name System[J]. *ACM SIGCOMM Computer Communication Review*, 1995, 25(1): 112-122.

[2] Zhang Y, Xia Z D, Fang B X, et al. An Autonomous Open Root Resolution Architecture for Domain Name System in the Internet[J]. *Journal of Cyber Security*, 2017, 2(4): 57-69.
(张宇, 夏重达, 方滨兴, 等. 一个自主开放的互联网根域名解析体系[J]. *信息安全学报*, 2017, 2(4): 57-69.)

[3] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system[J]. *Decentralized Business Review*, 2008: 21260.

[4] Name coin official website[EB/OL]. https://www.namecoin.org/. 2020.12/2020.12.

[5] Kalodner H A, Carlsten M, Ellenbogen P, et al. An Empirical Study of Namecoin and Lessons for Decentralized Namespace Design[A]. 14th Annual Workshop on the Economics of Information Security[C]. 2015:1-23.

[6] Namecoin. Namecoin Core integration/staging tree[DB/OL] https://github.com/namecoin/namecoin-core,2022.2/2022.3.

[7] Ali M, Shea R, Nelson J, et al. Blockstack: A new decentralized internet[J]. *Whitepaper, May*, 2017:1-22.

[8] Who-owns-the-ens-rootnode-what-powers-does-that-grant-them. [DB/OL]https://docs.ens.domains/frequently-asked-questions, 2022.3/2022.5.

[9] Praitheeshan P, Pan L, Yu J S, et al. Security Analysis Methods on Ethereum Smart Contract Vulnerabilities: A Survey[EB/OL]. 2019: arXiv: 1908.08605. https://arxiv.org/abs/1908.08605.pdf.

[10] InterNIC,DNS root zone file [DB/OL] https://www.internic.net/domain/root.zone, 2022.3/2022.3.

[11] Van Rijswijk-Deij R, Chung T, Choffnes D, et al. The Root Canary: Monitoring and Measuring the DNSSEC Root Key Rollover[C]. *The SIGCOMM Posters and Demos*, 2017: 63-64.

[12] Ongaro D, Ousterhout J. In Search of an Understandable Consensus Algorithm[C]. *The 2014 USENIX conference on USENIX Annual Technical Conference*, 2014: 305-320.

[13] Nelson J, Ali M, Shea R, et al. Extending existing blockchains with virtualchain[A] Workshop on distributed cryptocurrencies and consensus ledgers[C]. 2016.

[14] Blockstack document[DB/OL]. https://docs.blockstack.org/build-apps/references/bns. 2020.12/2020.12.

[15] Ali M, Nelson J, Shea R, et al. Blockstack: A Global Naming and Storage System Secured by Blockchains[C]. *The 2016 USENIX Conference on Usenix Annual Technical Conference*, 2016: 181-194.

[16] Ethereum naming service document. https://docs.ens.domains/. 2020.12/2020.12.

[17] Lu H, Jin C J, Helu X H, et al. DeepAutoD: Research on Distributed Machine Learning Oriented Scalable Mobile Communication Security Unpacking System[J]. *IEEE Transactions on Network Science and Engineering*, 2022, 9(4): 2052-2065.

[18] Wood G. Ethereum: A secure decentralised generalised transaction ledger[J]. *Ethereum project yellow paper*, 2014, 151(2014): 1-32.

[19] Hyperledger fabric sdk document [DB/OL]. https://hyperledger.github.io/fabric-sdk-node/release-1.4/tutorial-sign-transaction-offline.html. 2020.12/2020.12.

[20] Androulaki E, Barger A, Bortnikov V, et al. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains[C]. *The Thirteenth EuroSys Conference*, 2018: 1-15.

[21] Kang J W, Xiong Z H, Niyato D, et al. Incentivizing Consensus Propagation in Proof-of-Stake Based Consortium Blockchain Networks[J]. *IEEE Wireless Communications Letters*, 2019, 8(1): 157-160.

[22] Vukolić M. The Quest for Scalable Blockchain Fabric: Proof-of-Work Vs. BFT Replication[C]. *International Workshop on Open Problems in Network Security*, 2016: 112-125.

[23] Tian Z H, Li M H, Qiu M K, et al. Block-DEF: A Secure Digital Evidence Framework Using Blockchain[J]. *Information Sciences*, 2019, 491: 151-165.

[24] Zahnentferner J. Chimeric ledgers: Translating and unifying UTXO-based and account-based cryptocurrencies[J]. *Cryptology ePrint Archive*, 2018.

[25] Benet J. Ipfs-content addressed, versioned, p2p file system[J]. arXiv preprint arXiv:1407.3561, 2014.

[26] Castro M, Liskov B. Practical byzantine fault tolerance[A] 3rd OSDI [C]. 1999: 173-186.

[27] Schnorr C P. Efficient Signature Generation by Smart Cards[J]. *Journal of Cryptology*, 1991, 4(3): 161-174.

[28] Maxwell G, Poelstra A, Seurin Y, et al. Simple Schnorr Multi-Signatures with Applications to Bitcoin[J]. *Designs, Codes and Cryptography*, 2019, 87(9): 2139-2164.

[29] Bi W, Jia X Y, Zheng M L. A Secure Multiple Elliptic Curves Digital Signature Algorithm for Blockchain[EB/OL]. 2018: arXiv: 1808.02988. https://arxiv.org/abs/1808.02988.pdf.

[30] An alternative full node bitcoin implementation written in Go [DB/OL] https://github.com/btcsuite/btcd. 2020.12/2020.12.

[31] Johnson D, Menezes A, Vanstone S. The Elliptic Curve Digital Signature Algorithm (ECDSA)[J]. *International Journal of Information Security*, 2001, 1(1): 36-63.

[32] Mendel F, Nad T, Schläffer M. Improving Local Collisions: New Attacks on Reduced SHA-256[C]. *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2013: 262-278.

[33] Liardet P Y, Smart N P. Preventing SPA/DPA in ECC Systems Using the Jacobi Form[M]. *Cryptographic Hardware and Embedded Systems — CHES 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001: 391-401.

[34] Mockapetris P V. Domain names-concepts and facilities[J]. *Request for Comments 1034*, 1987: 1-55.

[35] Namecoin. Ncdns [DB/OL] https://github.com/namecoin/ncdns, 2020.1/2022.3.

[36] Iana, root zone database [DB/OL] https://www.iana.org/domains/root/db, 2020.12/2020.12.

[37] Tradeblock, bitcoin historical data [EB/OL]https://tradeblock.com/bitcoin/historical/6h-f-tsize_per_avg-01101, 2020.12/2020.1.

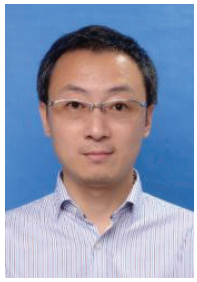
邓锦禧 于 2018 年在华南农业大学计算机科学与技术专业取得学士学位。现在在广州大学网络先进技术研究院计算机技术专业攻读硕士学位, 主要研究方向包括区块链安全等。Email: dengjx1160@gmail.com

苏申 哈尔滨工业大学博士。广州大学网络空间先进技术研究院副教授, 广州大学-奇安信云安全联合实验室常务副主任, 主要研究方向包括区块链安全、DNS 安全、路由安全、车联网安全等。Email: sushen@gzhu.edu.cn

李爽 于 2019 年在哈尔滨工业大学信息安全专业获得学士学位, 现攻读广州大学计算机技术专业硕士学位, 研究兴趣包括区块链和计算机网络安全等。Email: 2111906051@gzhu.edu.cn

韩毅 广州大学网络空间先进技术研究院研究员, 主要研究方向包括个人隐私保护、数据安全、社交网络分析等。Email: hanyi@gzhu.edu.cn

郭泽宇 于 2017 年在西安电子科技大学软件工程专业获得学士学位。现在广州大学网络空间先进技术研究院电子信息专业攻读硕士学位, 主要研究方向包括区块链安全等。Email: 2112006095@gzhu.edu.cn



田志宏 哈尔滨工业大学博士。教授, 博士生导师, 广州大学网络空间先进技术研究院院长; 广东省“珠江学者”特聘教授。长期致力于网络攻防对抗、网络靶场、主动实时防护等网络空间安全热点领域的研究工作。Email: tianzhihong@gzhu.edu.cn