## 第四章 基于区块链的访问控制模型

### 摘要

研究了互联网中一个关键的访问控制问题。通过分析访问控制在区块链上的应用原理，我们提出了一个基于智能合约的框架，该框架由管理设备和合约的管理合约（MC）、用于控制访问状态的访问控制合约（ACC）以及用于动态访问控制的信誉合约（RC）组成，以实现链上由信誉驱动的动态访问控制。通过基于智能合约的政策执行，MC能够进行用户和用户属性的细化分层管理；通过接口实时添加更新主体和资源之间的访问策略，实现访问策略的规范化管理。ACC在细粒度属性访问控制的基础上融合动态访问控制思想，针对用户行为与静态属性，设计了一套将其综合量化的信誉算法。该信誉算法分为正面影响和负面影响，当用户出现不当行为或者用户属性错误导致的访问策略检查失败，信誉值会随不当行为的严重程度而有所下降；反之，当用户自我修正或检测后，连续进行正常访问，也会对此进行信誉值的奖励。最后将信誉值反映到区块阻塞程度上，当出现区块阻塞后，在该特定阻塞范围内将禁止来自该用户的访问，直至阻塞结束。由于信誉算法存在惩罚和奖励，所以能够更好地规范用户行为。为了展示该框架的应用，本文在Quorum区块链中使用智能合约实现了该访问控制系统，并进行了相应的性能测试与动态行为检验测试。实验表明，该访问控制框架能够有效识别异常节点，并阻止异常节点的连续访问。

### 4.1 问题描述

近年来，随着互联网的快速发展和相关技术的进步，我国将加速信息化的发展放在了突出位置。科技作为国家发展的重要方面，构建安全有效的科技管理信息平台是国家推进信息化发展战略的主要工作之一。在科技管理平台中，其主要工作需要收集、存储、处理和共享用户信息和科技研究信息，并实现科研人员、单位和部门之间的信息互联互通和资源共享，以此促进信息管理的精细化和智能化，从而提高科技信息管理工作的规范化程度和工作效率。在一定程度上，科技管理信息平台加快了用户数据和科技研究数据在网络系统中的流动，为相关单位和部门的信息管理工作带来便利。然而，由于科技管理平台中所存储和共享的科技数据是信息时代最基本、最具活力、最具影响力的技术资源和战略资源，对社会发展有着深远影响，因此，科技管理信息平台所面临的数据安全威胁更为突出。在科技管理信息平台的数据信息流通和共享过程中，由于科技管理平台不仅要对不同来源的共享数据进行统一管理，同时还需要提供不同层次用户间的信息资源交流和共享，但可能因某些恶意应用或者用户违规操作造成数据非授权访问、越权访问、数据泄露等安全问题，不仅对用户自身数据安全造成影响，甚至在一定程度上会给国家安全带来严重危害。因此，如何在科技管理信息平台中实现最大程度信息共享的同时保证敏感数据不被非法访问是一项具有挑战的重要任务。

目前，针对数据共享安全和授权访问的研究主要集中于访问控制技术。访问控制技术是维护信息数据安全、保护隐私信息、实现数据共享安全的基础和先决条件，是解决用户访问资源数据安全问题核心技术之一。它可根据安全策略管理所有资源的访问请求，并确定资源权限的访问请求是否符合规则，从而有效防止非法用户访问系统资源和合法用户非法使用资源，达到保护系统安全存储和处理数据信息的目的。随着计算机技术的更新迭代，学者先后提出了多种访问控制模型，例如基于对象的访问控制（Object-based Access Control, OBAC）[1]、基于任务的访问控制（Task-based Access Control, TBAC）[2-4]，基于角色的访问控制（Rolebased access control, RBAC）[5-7]、基于属性的访问控制（Attribute-based access control, ABAC）[8-14]等。这些策略大多通过集中式方法授予用户权限，这种方式缺乏透明性，易产生单点故障问题。同时，由于科技管理平台所涉及范围广、使用者类型多，不同类型使用者有不同的权限访问需求，因此，以上访问机制在科技管理平台中不能很好地适用。

在科技管理平台下，其访问控制机制在安全性、适用性以及动态性方面的要求更高，同时也需要提高策略判定结果的准确性，以保证数据不会被泄露。此外，还需细粒度区分用户类别以满足不同类别用户的复杂访问需求。因此，为满足科技管理平台下的访问控制机制的需求，本文拟结合基于属性的访问控制以及区块链技术来保证访问控制机制的灵活性、细粒度以及安全性。针对基于区块链的访问控制，学者先后进行了如下研究，例如利用智能合约实现基于属性的访问控制模型完成访问控制[15-17]以及基于区块链的大数据访问控制研究[18]等等。

基于属性的访问控制是基于实体属性而不是用户身份来判断是否允许用户对请求资源的访问。它通过属性描述实体和约束以达到严格控制用户授权的各种条件，精确设定属性-权限关系，实现最小权限原则。同时在授权决策过程中基于访问主体和资源的属性，可根据访问需求进行设置，实现了策略管理和权限判定的分离，具备更强的灵活性和扩展性。区块链技术是将数据区块按照时间顺序相连组合的链式数据结构，并以密码学方式保证的不可篡改和不可伪造的分布式账本。它采用协商一致的规范和协议，无需借助第三方介入就可使链中所有节点达成共识，具备数据难以篡改和去中心化两大核心特点，解决了信任与价值可靠传递的问题，很好地维护了数据完整性、可信性和安全性。

综上所述，本文从科技管理平台的访问控制需求出发，拟结合基于属性的访问控制机制和区块链技术的优势，提出一种信誉驱动的动态访问控制模型，通过建立用户本身、请求的操作和资源敏感程度等属性计算用户信誉，通过信誉对控制策略进行调整，以此建立科技管理平台中高效、安全的访问控制机制，增强访问控制机制的动态性、安全性。

### 4.2需求分析

在科技管理平台所在的网络环境下，由于其使用者类型众多，分布较为分散，因此，其访问控制机制在安全性、适用性以及动态性方面的要求更高，同时也需要提高策略判定结果的准确性，保证数据不会被泄露。此外，还需细粒度区分用户类别以满足不同类别用户的复杂访问需求，保证各类用户对资源的访问权限更加精准。综上，科技管理平台下的访问控制机制应具备以下三大特征：

（1）动态授权：科技管理平台所使用情况较多，其访问控制机制应时刻关注用户请求状态，并根据状态予以不同权限，实现灵活动态授权；

（2）细粒度定位用户：科技管理平台所面向用户类型多，访问广，其访问控制机制应对不同类型用户能够更加精准描述，以保证细粒度区分访问权限；

（3）高安全性。科技管理平台中的数据及其关键和敏感，其访问控制机制应在数据共享安全性中有更高的要求。

在传统的访问控制技术中，每种访问控制模型都具备不同的特点，例如：自主访问控制和强制访问控制模型，其授权规则简单，但难以实现动态授权，不够灵活；基于角色的访问控制易于拓展，但无法动态更改角色关系，无法很好地满足上述要求。

基于属性的访问控制是基于实体属性而不是用户身份来判断是否允许用户对请求资源的访问，具有属性可变、具备连续决策的特点。它通过属性描述实体和约束以达到严格控制用户授权的各种条件，精确设定属性-权限关系，实现最小权限原则。同时在授权决策过程中基于访问主体和资源的属性，可根据访问需求进行设置，实现了策略管理和权限判定的分离，具备更强的灵活性和扩展性。该方法弥补了一些机制方法的不足，能够较好适用于科技管理平台下的访问控制机制要求，所以本文将其作为科技管理平台下的访问控制基础模型。

但以上这些传统方法大多都是采用集中式方法对用户进行授权，即所有信息都存储在服务器上，同时所有处理过的数据都集中在服务器上，这意味着提供商可能会获得对其客户端的数据进行未经授权的访问。因此，如果将管理平台访问控制中的安全业务外包给第三方，就必须对他们给予足够的信任。另外，传统访问控制的另一个巨大缺陷是，它易受到各种攻击，在某些情况下很容易被破坏。而区块链技术是将数据区块按照时间顺序相连组合的链式数据结构，并以密码学方式保证的不可篡改和不可伪造的分布式账本。它采用协商一致的规范和协议，无需借助第三方介入就可使链中所有节点达成共识，具备数据难以篡改和去中心化两大核心特点，解决了信任与价值可靠传递的问题，很好地维护了数据完整性、可信性和安全性。区块链技术的去中心化特性以及提供的端到端的安全性和加密，很大程度上消除了人为错误的风险，并能够很好地防止黑客攻击。因此，将区块链与传统访问控制相结合，能够消除传统访问控制模型中存在的集中式问题，同时减少其被攻击的风险，大大提高访问控制机制的稳定性和安全性。

但由于在科技管理平台中会出现恶意的行为（如短时间频繁的调用）导致产生过量的交易，从而降低合法交易被区块链收集的概率，或者使得确认时间延长，而区块链本身无法对这类行为做出规避。因此，针对该问题，本文以传统的基于属性的访问控制为基础，结合区块链技术，设计了基于信誉驱动的动态访问控制机制，其核心思想是基于操作的严重性、资源的敏感性、用户的访问历史记录等信息，对决策做出动态的调整，从而完成对恶意行为的惩罚和合法行为的奖励，以规避区块链为基础的访问主体的恶意行为。为了对访问主体的恶意行为有个良好的解决措施，本文引入信誉算法，同时由于不同的恶意行为所造成的严重程度不一致，因此不仅要对访问者错误行为做出惩罚，还要对访问者自我检测、自我更正能力做出一定的奖励和容忍，通过信誉值来对访问者进行访问权限的约束，实现访问权限的动态变化，以提高访问控制机制的灵活性和安全性。

### 4.3信誉驱动的动态访问控制系统设计

#### 4.3.1系统架构

根据上述基于区块链的访问控制模型存在的问题，以及现有的相关工作基础，本文针对图4.1所示的科技信息管理平台系统，在区块链的技术基础上提出信誉驱动的动态访问控制模型，在科技管理信息平台中实现最大程度信息共享的同时保证敏感数据不被非法访问。

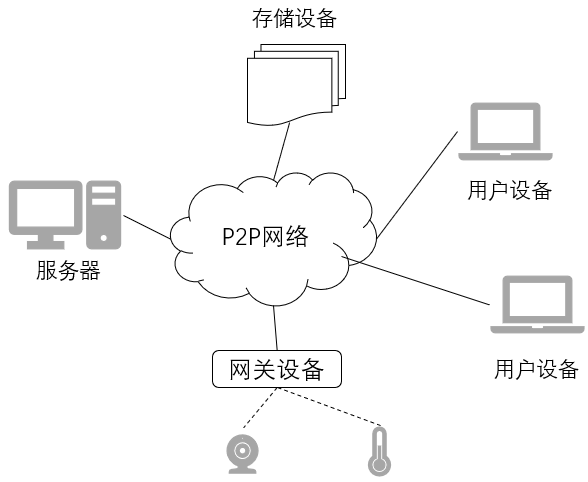


图4.1信息管理平台

本文考虑的科技信息管理平台系统由大量服务器、存储设备、网关和用户设备组成，这些用户与设备通过点对点网络连接在一起。系统中还存在许多其他通信设备，这些设备通过网关连接到P2P网络。该系统所含对等体的主要作用解释如下：

服务器是一种设备或设备集群，可以与用户和设备进行交互，为用户提供各种服务。

存储设备可以为系统的其他对等体存储数据，如服务器的应用数据、传感器采集的环境数据、用户配置文件等。

用户设备是指用户可以享受服务器提供的服务，并对存储设备进行读写的设备。

网关是通过蓝牙、Wi-Fi、Zigbee等短距离通信技术，将一群设备连接到P2P网络上，同时充当这些通信设备的业务代理。

针对上述科技信息管理平台，本文提出如下访问控制模型设计思路：

首先，模型以在区块链上实现的访问控制为背景，集成区块链技术与访问控制技术的优势，将功能封装在智能合约中，部署在节点私链上，实现信誉驱动的动态访问控制模型的一系列功能。

第一部分是传统的基于属性的访问控制功能实现，通过基于智能合约的政策执行，规范用户和用户属性的细化分层管理；再通过接口实时添加更新主体和资源之间的访问策略，实现访问策略的规范化管理。

第二部分则是在第一部分的基础上融合动态访问控制思想，针对用户或设备的行为，设计一套将行为量化的信誉算法。该信誉算法分为正面影响和负面影响，当用户出现不当行为或者用户不在服务范围导致的访问策略检查失败，信誉值会随不当行为的严重程度而有所下降；反之，当用户自我修正或检测后，连续进行正常访问，也会对此进行信誉值的奖励。最后将信誉值反映到区块阻塞程度上，当出现区块阻塞后，在该特定阻塞范围内将禁止来自该用户的访问，直至阻塞结束。由于信誉算法存在惩罚和奖励，所以能够更好地规范用户行为。

综上，该模型的贡献主要有以下方面：

1.提出一个新的系统，集成了区块链和智能合约功能。

2. 通过基于智能合约的政策执行，以规范用户和设备的资源访问。

3.提出信誉服务，基于每个访问结果建立主体声誉，奖励(正面声誉)或惩罚(负面声誉)对客体资源的不当行为，从而帮助更好规范用户和设备的行为。

4. 以太坊客户端配置使用IBFT作为共识协议，以消除节点的计算负担。

下面介绍一下该模型的系统架构，如图4.2所示。

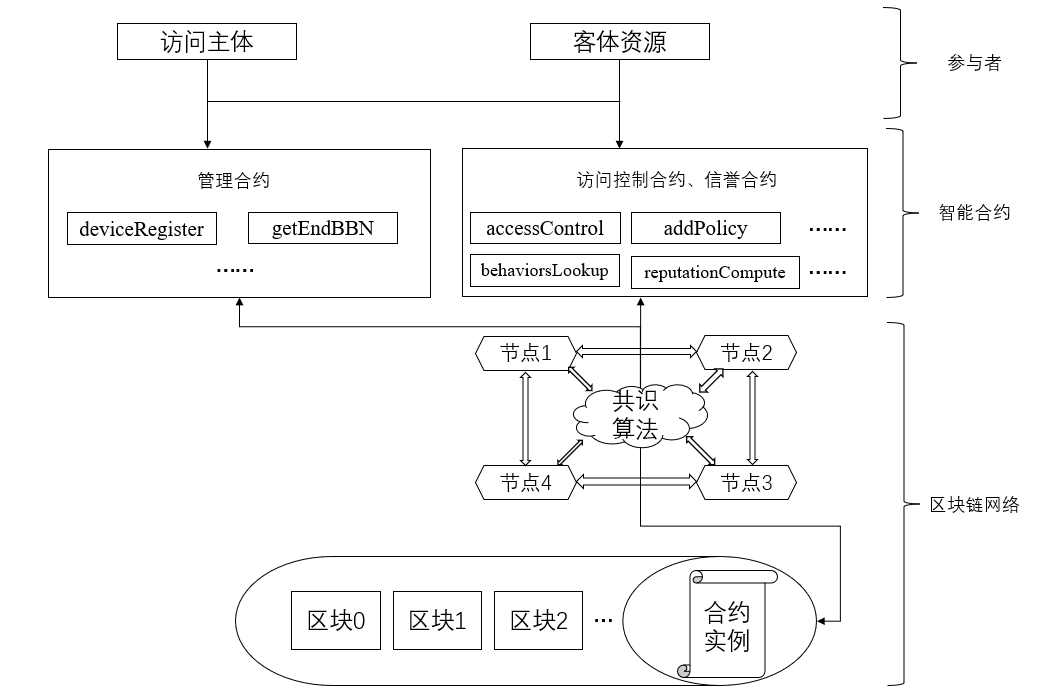


图4.2访问控制系统模型

（1）模型的简化架构包括三个主要部分：参与者、智能合约和区块链网络。

参与者：访问主体和客体资源，即上文提到的服务器、用户设备、存储设备及其他通信设备等，这些主体和客体，要么是验证节点要么是轻量级节点。验证节点主要负责以下工作：

* 存储区块链副本。
* 通过创建、提议和添加区块到区块链来验证交易并参与共识过程。

轻量级节点只是发出事务，并依赖于验证器来添加和验证他们的事务。此外参与者通过部署的智能合约ABIs来与区块链通信。

（2）智能合约：基于区块链的访问控制基础上，智能合约尤为重要。为了能够解决上述互联网系统的关键访问控制问题，本文提出一个基于智能合约的访问控制框架，将具体功能集成到智能合约之上，并部署到区块链上实现访问控制功能。该系统中包括管理设备和合约的管理合约、用于控制访问状态的访问控制合约以及用于动态访问控制的信誉合约。

（3）区块链网络：即带有智能合约的分布式账本，它由P2P网络和管理网络通信的共识协议组成。将不同类型的交易与合约安排到网络中，该网络由许多计算机组成，但是在没有整个网络的共识的情况下无法更改数据，从而实现去中心化的安全与信任。

#### 4.3.2智能合约平台

##### 4.3.2.1以太坊平台

提出的框架基于以太坊智能合约平台，通过该平台，系统的每一个对等体都可以对其资源进行访问控制。以下简要介绍以太坊平台的主要元素。

帐户/地址：以太坊有两种类型的帐户：外部控制帐户和合约帐户，都由一个20字节的地址标识。在本文中，前者称为帐户，后者称为智能合约。

智能合约：智能合约是一个包含代码（即其功能）和数据（即其状态）的特殊帐户。代码采用特定于以太坊的二进制格式（即以太坊虚拟机字节码），并由帐户部署到名为区块链的全局数据库。智能合约通常提供许多功能或应用程序二进制接口(即ABIs)，可以用来与之交互。这些ABI可以通过从帐户发送交易或从另一个合同发送消息来执行，它们也可以通过简单地调用函数而不发送事务和消息来执行。

交易和消息：交易是一个帐户签名并发送的数据包，以将一些以太坊(以太坊的原生令牌)转移到另一个帐户。除了传输以太，交易也可以发送一些参数来执行合同的ABIs。消息类似于交易，但它是由一个合约而不是一个帐户发送的，以运行另一个合约的关联ABI。

Quorum：选择Quorum平台搭建节点私链，在这个区块链的基础设施之上，部署并运行本文设计的智能合约实现访问控制。Quorum是一个企业区块链平台。它是公共以太坊客户端“geth”的一个分支，具有多个协议级别的增强功能，以支持业务需求。从根本上说，Quorum是增强了企业功能的公共以太坊客户端。它在许可网络中提供隐私功能、企业许可和改进的性能。Quorum架构如图4.3所示。

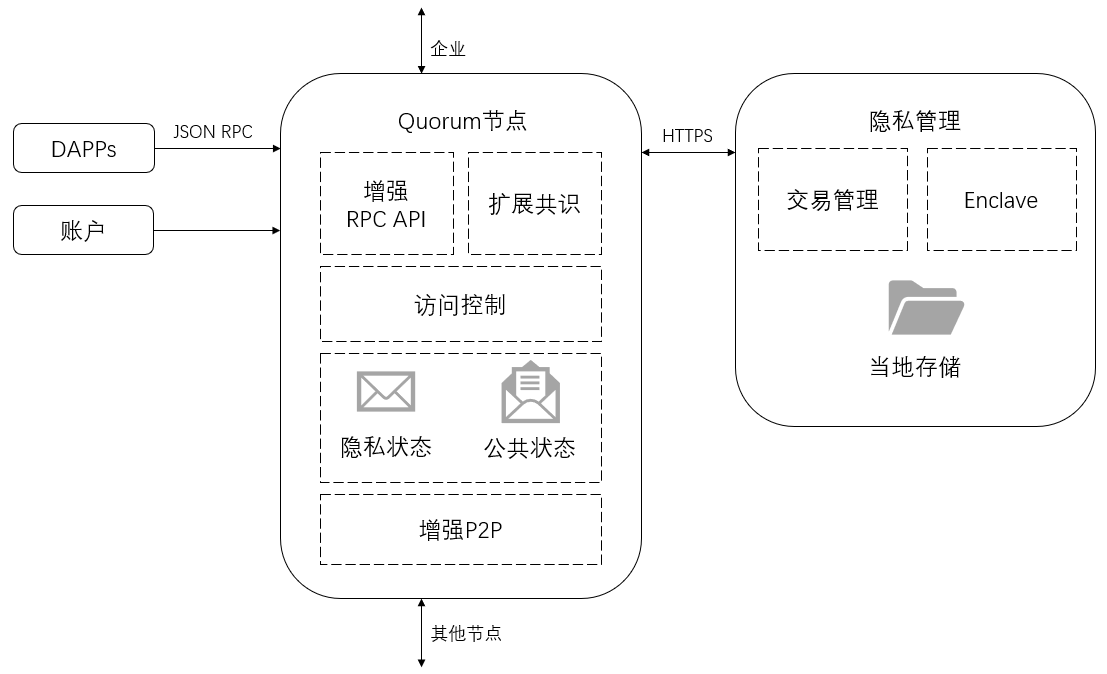


图4.3Quorum结构

挖矿：挖掘是一个将新的事务块和合约块包含到区块链中的过程。如果交易和合约的格式以及被调用的ABIs的结果是有效的，其他节点将把新的块包含到它的本地区块链；否则，他们将丢弃该块。通过挖掘，整个系统在区块链上达成了一个共同的抗篡改共识，任何参与者都不能通过错误地执行ABI来欺骗其他参与者，只要它控制的计算能力不超过系统的一半。这也是实现系统可信访问控制的关键。

##### 4.3.2.2系统配置

为了在设计的访问控制框架中应用以太坊平台，需要对系统进行以下基本配置：

每个对等体必须与以太坊帐户相关联，以在系统中表示自己。使用该帐户，每个对等方都可以声明智能合约的部署，并在访问控制期间识别自己。

以太坊客户端可以在系统中除物联网设备外的所有对等点上运行。假设所有客户端在同一个块上同步。通过客户端，除物联网设备外的其他对等端都可以直接与区块链交互，部署智能合约，发送交易来运行智能合约的ABI。这些对等点还可以充当矿工，为系统执行挖掘任务。

对于某些通信设备没有以太坊客户端，网关作为其本地设备的代理，对这些设备的资源进行访问控制。为了实现这一目标，每个网关存储其本地设备的帐户，并使用这些帐户签署交易，代表其本地设备部署和运行智能合约。

#### 4.3.3访问控制框架

##### 4.3.3.1框架流程

访问控制必须由所有资源所有者实现，以防止未经授权使用其资源。例如，服务器必须能够阻止来自未注册用户的访问请求，或来自已注册用户的未订阅的某些服务的访问请求。为了防止非法使用存储空间和数据，存储设备必须能够限制非法的对等体对数据的查询或存储请求。外围设备也必须能够拒绝未经授权的访问请求，以检索其数据或控制其执行器。本文的目的是解决上述系统的关键访问控制问题，提出一个基于智能合约的访问控制框架，根据合约具体函数实现信誉驱动的动态访问控制功能。该访问控制框架流程如图4.4所示。

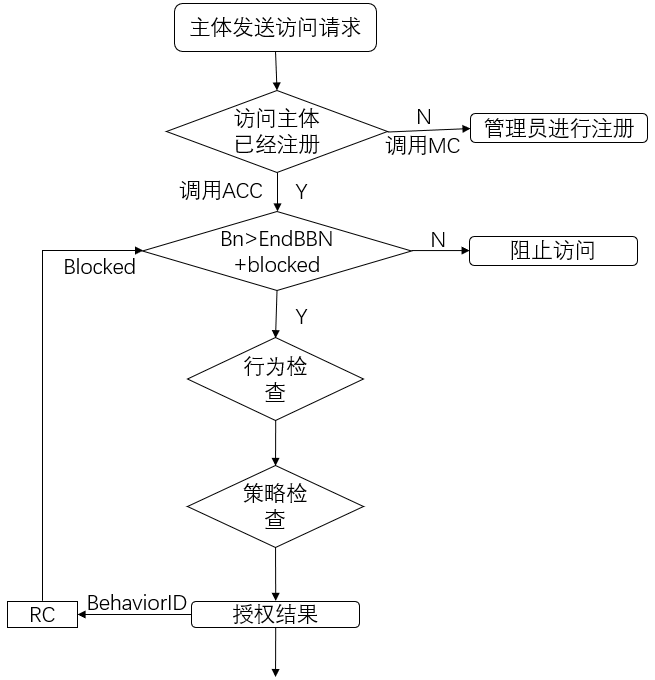


图4.4访问控制流程图

主体发送访问请求，调用管理合约MC，查看该访问者是否已经注册。如果没有注册，则通过MC接口注册用户或设备；如果已经注册完成，调用ACC进行访问控制。然后，首先查看访问者的区块数是否已经超出阻塞区块号，如果没有，则直接阻止该区块访问内来自该用户的所有访问。如果注册区块号在阻塞范围外，则依次进行行为检查和策略检查。行为检查主要检查是否频繁访问，策略检查主要分为策略检查失败和重要策略检查失败，根据不同的检查结果给出BehaviorID以及访问结果。ACC判断所得BehaviorID转发给信誉合约RC，作为信誉计算的输入，通过计算，输出该访问主体的阻塞区块数，返回给ACC，及时更新。

本节介绍基于智能合约的分布式访问控制框架。之后介绍框架中的智能合约系统，最后简要说明框架提供的主要功能。

##### 4.3.3.2智能合约系统

为实现上述框架功能，本文设计以下三种合约：管理合约MC（ManagementContract），访问控制合约ACC（AccessControlContract）和信誉合约RC（ReputationContract），这几种合约间的调用关系如图4.5所示,该框架包含多个访问控制合约（ACCs），每一个都实现了一对对等体的访问控制。一个信誉合约（RC），从ACC接收行为结果计算信誉值，并确定相应的惩罚。一个管理合约（MC），用于用户注册和管理其他合约。

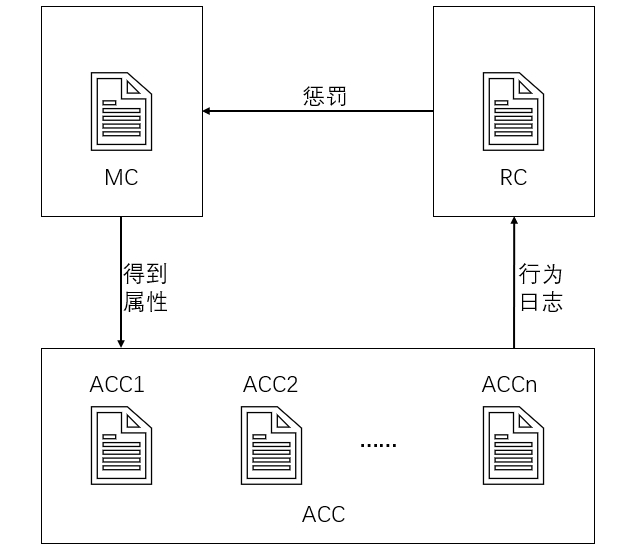


图4.5智能合约调用关系

##### 1.访问控制合约

访问控制合约ACC负责管理资源属性、访问控制策略和执行访问控制决策。一个ACC是由一个对等体（客体）部署，它用来控制来自另一个对等体（主体）的访问请求。本文假设主体-客体对可以对多种访问控制方法达成一致，每个方法由一个ACC实现。因此，一个主体-对象对可以与多个ACC相关联，但一个ACC能且只能与一个主体-对象对相关联。在这个框架中，为了控制来自主体的访问请求，每个ACC不仅通过检查预定义的策略实现静态访问权限验证，还通过检查主体的行为实现动态验证。对ACC相关函数的调用，其行为日志会提交给RC供分析。在执行访问控制决策时，会首先从MC读取访问者的endBBN字段，查看是否大于当前区块号，如果大于则直接阻止该请求，不会进入区块链网络。

为了实现访问控制，ACC维护了属性策略和行为策略列表，如表1与表2所示。

表1 ACC属性策略

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| resource | action | attrOwner | attrName | operator | attrValue | importance |
| location | read | subject | deviceType | = | gateway | 2 |
| basicInformation | read | subject | deviceID | = | pallat23 | 0 |
| …… |  |  |  |  |  |  |

每行的基本字段如下所示：

resource：访问客体。

action：访问行为。

attrOwner：属性所有者，主体还是资源。

attrName：属性名称。

operator：条件操作符。

attrValue：需要满足的条件。

importance：该策略的重要等级。当信誉值不为0时，它将被提交给RC计算。

表2 ACC属性策略

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| resource | action | ToLR | NoFR | minInterval | threshold | algorithm |
| location | read | 2022.7.12.10:50:14 | 3 | 10 | 5 | deny |

每行的基本字段如下表示：

resource：访问资源。

action：访问行为。

ToLR：最后一次访问时间。

NoFR：短时间内频繁请求次数。

minInterval：两次请求之间最小的允许间隔（以s为单位）

threshold：NoFR的阈值，超过该值怀疑有不当行为。

algorithm：确定规则冲突、允许、拒绝三种情况的结果。

ACC提供以下主要ABI来管理策略和实现访问控制。

accessControl（）ABI设计思路如下：

访问主体发送访问请求，调用MC检查访问主体是否已经注册，如果成功注册，则开始执行访问控制流程。在访问控制流程中，需要检查访问者当前注册区块数是否大于最后阻塞区块数与阻塞区块数的和，如果小于，则终端访问；如果大于，进行后续访问控制程序检查。在此过程中，首先进行该访问主体的行为检查，计算区块当前时间与最后一次访问时间的差值是否大于最小允许时间，如果大于，则进行策略检查；反之频繁访问次数则进行叠加，次数越多，恶意行为越明显，惩罚措施阻塞区块数越多。最后进行策略检查，如果策略已经定义，则检查计算属性值与当前属性值是否符合预定义的操作符关系。策略检查主要有策略检查失败和重要策略检查失败，根据行为检查与策略检查的不同情况综合得出不同的BehaviorID，转发给RC，进行信誉计算，得出EndBBN，再返回给ACC进行访问控制。该函数实现过程如下算法1所示：

|  |
| --- |
| **算法1：访问控制函数** |
| **输入:**resource, action  **输出:**finalResult, behaviorID, bn   1. **if** endBBN<bn **then** 2. 用户仍然被阻塞 3. **end** 4. 频繁请求行为检测 5. **if** timestamp-ToLR≤minInterval 6. NoFR++ 7. **if** NoFR≥threshold **then** 8. behaviorCheck←true 9. **end** 10. **else** 11. NoFR←0 12. **end** 13. 更新最后一次请求时间 14. 策略检查 15. **if** policieslength = 0 **then** 16. 策略未定义 17. **End** 18. **for** i = 0 to length **do** 19. current←policies[resource][action][i] 20. **if** 属性所有者为访问主体 **then** 21. **if** 包含deviceID and deviceTyp and deviceRole **then** 22. attrValue为固有属性 23. e**lse** 24. attrValue为自定义属性 25. **End** 26. **else** 27. attrValue为资源属性 28. **end** 29. **if** 满足> ，并且attrValue≤current 30. currentPolicy←true 31. **if** 满足<，并且 attrValue≥current 32. currentPolicy←true 33. **if** 满足=，并且attrValue=current 34. currentPolicy←true 35. **If** currentPolicy存在 then 36. result[1]++ 37. **if** 安全等级≠0 **then** 38. result[2]++ 39. **end** 40. **else** 41. result[0]++ 42. **end** 43. **end** 44. **if** behaviorCheck and not policyCheck **then** 45. behaviorID←1 46. **if** not behaviorCheck and policyCheck**then** 47. **if** result[2]>0 **then** 48. behaviorID←3 49. **else** 50. behaviorID←2 51. **end** 52. **end** 53. **if** policyCheck or behaviorCheck **then** 54. 拒绝访问 55. **else** 56. 同意访问 |

该ABI接收访问控制所需的信息，并返回访问结果和惩罚。当主体调用该ABI进行访问授权时，首先会对比EndBBN字段，然后开始静态属性策略检查和动态行为检查。策略结果包括：策略检查成功、策略检查失败、重要的策略检查失败；行为结果包括：访问授权、频繁访问、策略检查失败、重要的策略检查失败。检查结束后，返回结果包括：主体、最后结果、行为ID、区块数。

addPolicy（）接收输入信息，并添加到策略表中。并且只允许管理员添加策略。

addResourceAttribute（）为方便管理用户属性，当用户已经完成注册的情况下，如果想要对其属性进行更新，可以调用该API。

deleteACC、deletePolicy（）、deletePolicyItem（）、deleteResourceAttr（）可以用来删除访问控制合约、属性以及策略等。

getPolicy（）、getPolicyItem（）、getResourceAttr（）、manager、mc、rc等可以用来检索策略、属性以及合约地址等。

stringCompare（）该函数用来判断两个数是否相等，使用长度加哈希进行比较，减少gas消耗。

stringToUnit（）是一个实用函数，将字符串转换成unit。

updateEnviroment（）用来更新计算环境，包括两次请求之间最小的允许间隔、NoFR的阈值以及确定规则冲突、允许、拒绝三种情况的结果算法。

updateManager（）、updateResourceAttr（）、updateSCAddr（）用来更新管理员、资源属性以及合约地址等。

##### 2.管理合约

管理合约MC负责管理其它合约和管理用户或者设备属性。ACC在进行访问控制决策时，会首先从MC获取访问者的相关属性，然后和定义的策略进行匹配。在MC中注册的所有用户都新增了一个endBBN固定属性，用于设置终止阻塞的区块号，该字段只能被信誉合约更新，用户和管理者都无法改动。

为方便管理其他合约和用户注册，管理合约主要提供以下ABI进行合约和用户管理。

deleteDevice（）deviceRegister（）用户设备注册与删除，主要包括以下信息：用户、管理者、与用户相关的合约、ID、类型、角色。此类ABIs功能设计如下：

在进行注册之前，首先检查用户是否已经注册，如果有，则不允许重复注册；如果没有，随后检查注册者是否为该用户的管理员，只有管理员才能进行注册。该函数实现过程如算法2所示：

|  |
| --- |
| **算法2：用户注册** |
| **输入: address and manager and ACC; ID; Type; Role; bn**  **输出: Registered**   1. 初始化 2. **if** 用户未注册 3. **if** 注册者非管理员 4. 只有管理员可以注册 5. **end** 6. lookupTable[device].variate← data 7. isvalued← true 8. isACCaddress← true 9. **end** |

addAttribute（）、deleteAttribute（）这是最基础的属性API，ACC中的属性管理是针对客体资源，MC中的管理合约则针对访问主体。

getCustomedAttribute（）用来获取自定义属性。

getEndBBN（）获得结束阻塞区块号，初始值为用户注册时的阻塞区块号。

getFixedAttribute（）获取固定属性。

isACCAddress用来判断一个地址是否为访问控制合约地址，由信誉合约使用。

setRC（）设置信誉合约。

updateAttribute（）、updateEndBBN（）更新访问属性，updateEndBBN只能由信誉合约调用。

updateManager（）、updateRC（）更新管理员以及RC。

##### 3.信誉合约

基于智能合约实现的系统中，基于属性的访问控制合约侧重用户静态属性，然而现实中恶意的行为（如短时间频繁的调用）可能产生过量的交易，从而降低合法交易被区块链收集的概率，或者使得确认时间延长，而区块链本身无法对这类行为做出规避。本文采纳动态访问控制的思想，基于操作的严重性、资源的敏感性、用户的访问历史记录等信息，对决策做出动态的调整，从而完成对恶意行为的惩罚和合法行为的奖励。采用信誉算法的原因是，不同的恶意行为严重程度不一致，所以不仅需要完成惩罚和奖励的基本功能，也需要提供一定的容忍。

信誉合约是ACC进行访问控制判断的基础，根据用户行为采取不同的奖惩力度。通过信誉算法设计，信誉合约RC根据ACC提交的行为记录计算信誉函数的值，并根据该值计算阻塞区块数，最后调用MC的相关函数更新对应的endBBN字段，做出惩罚。所以该固定属性只允许RC进行计算更新。

信誉合约最主要的就是信誉算法的思路设计，信誉算法的思路是信誉合约根据访问控制合约提交的行为记录计算信誉函数的值，并根据该值计算阻塞区块数，最后调用管理合约相关函数更新对应的EndBBN字段，做出惩罚。我们根据以上思路进行信誉算法设计。

用户i的信誉值由两部分组成，合法行为的正面影响和恶意行为的负面影响。公式如式（1）所示。

（1）

​为正面影响部分，​为负面影响部分，​和​分别是权重。

信誉值的负面影响部分​​也可以称为惩罚函数，其值与历史恶意行为的数量和类型有关，每个恶意行为的影响随着时间的推移逐渐减小，但不可以变为0，具体函数如式（2）所示。

（2）

其中​表示用户i当前的恶意行为总数，k表示历史恶意行为的索引顺序，​表示第k个恶意行为的惩罚系数,该系数在0-1内取值。k的最大值为​，因此分母最小值为1，不会导致值过大；k从0开始取值是为了保证所有历史恶意行为都能对当前结果产生影响。恶意行为种类包括如下三种：

短时间高密集的请求，行为 ID 为 1；

策略检查失败（非法访问），行为 ID 为 2；

重要策略检查失败。本文标识某条策略条目为重要，当违反该策略时，会作为恶意行为进行记录，比如，某个用户的访问目标一直都是固定的，突然发生了改变，说明可能是恶意的，又或者访问者的位置突然发生了改变等。这一类恶意行为优先级高于普通的策略检查失败（非法访问），计算时权重更高，行为 ID 为 3。

信誉值的正面影响部分​可以称为奖励函数，其值与合法行为数量正相关，定义如式（3）所示。

（3）

其中​是​的上界，防止奖励无限制积累，​为用户i的合法行为总数，​是上一次做出惩罚时计算的最后一个合法行为索引，因此，​其实相当于当前计算奖励值得一个滑动窗口大小。合法的行为只有一种：访问通过，​是合法行为的权重。

每一次行为提交都会更新行为列表，合法行为添加到合法行为列表，恶意行为添加到恶意行为列表，然后分别计算惩罚函数和奖励函数的值并得到最终的信誉值。当信誉值小于0时，会计算阻塞区块数并更新MC中用户对应的终止阻塞区块号属性，同时更新计算使用的最后一个合法行为索引，当下次统计合法行为时，该索引之前的合法行为将不会被再次计算。与之相对的，恶意行为记录永远不会清空，虽然随着时间的推移产生的影响变小，但不会消失，因此每次惩罚函数计算都会读取所有恶意行为。

惩罚手段是阻塞用户的访问请求，意思是计算一个阻塞区块数，从当前区块开始的这一段区块内，来自该用户的所有访问请求都被拒绝。阻塞时间计算如式（4）所示。

（4）

惩罚函数的值越大，阻塞时间越长。

根据以上信誉算法功能与设计思路，信誉合约主要函数ReputionCompute Function算法实现如算法3所示：

|  |
| --- |
| **算法 3: 信誉值计算函数** |
| **输入: subject; behaviorID; bn**  **输出: finalResult; bn; credit; Tblocked**   1. ACC合约调用 2. **if** behaviorID = 0 3. 行为日志添加为合法数组 4. **else** 5. 行为日志添加为非法数组 6. **end** 7. 初始化 8. **if** 非法行为 10. **end** 11. **if** 合法行为 13. **end** 14. **do** 16. **while**  and 计算完毕 17. **if** bn <endBBN and credit <0 and behaviorID ≠ 0 18. **if** legLen>begin 19. begin=legLen-1 20. **end** 21. TBlocked= 2−Cri 22. 更新endBBN 23. **end** 24. **if** behaviorID = 0 25. 认证授权, bn, credit, Tblocked 26. **else if** behaviorID = 1 27. 频繁访问, bn, credit, Tblocked 28. **else if** behaviorID **= 2** 29. 策略检查失败, bn, credit, Tblocked 30. **else if** behaviorID = 3 31. 重要的策略检查失败, bn, credit, Tblocked 32. **end** |

由于以太坊智能合约语言 Solidity 不支持浮点数定义和运算，因此公式中涉及的除法运算和浮点数定1义需要调整，本文导入ABDKMathQuad.sol合约辅助计算。该信誉合约主要提供以下ABI实现信誉计算。

behaviorsLookup表示行为被地址映射，输入地址可以查询行为。

evAttr表示影响该信誉值的环境因素。

reputationCompute（）计算信誉值，然后调用MC相关属性进行惩罚。

updateEnvironment（）更新信誉函数参数。注意，真正的输入值为value/base。

##### 4.3.3.3框架功能

在ACC、MC和RC的帮助下，该框架可以提供许多功能，方便该信息管理平台用户以及设备的访问控制。

1. 用户管理

主要包括注册与删除，通过管理合约实现。例如，用户注册，MC调用deviceRegister ABI，将管理者、地址、ID、类型、角色、区块号等信息添加到列表中。注意，只有管理者才可以进行注册，并且不允许重复注册。

1. 属性管理

包括addAttribute ABI添加属性、updateAttribute ABI更新属性、deleteAttributeABI删除属性。属性又划分为固定属性与自定义属性，固定属性包括ID、类型、角色以及endBBN，其中固定属性中的endBBN属性则是进行信誉判定的重要依据。为方便调用，MC合约主要管理访问主体属性，ACC主要管理访问客体属性。

1. 访问控制

首先验证endBBN，endBBN是通过信誉合约计算的结束阻塞区块号，如果主体bn＜endBBN小于区块数，则不允许访问。之后对访问主体进行行为检查和策略检索与决策。行为检查主要查看访问主体是否频繁请求；策略检测则主要根据主体提供的各种属性进行策略遍历，主要有策略检查成功、策略检查失败以及重要策略检查失败。最后将两种不同检测进行综合，给出最后的访问结果。具体步骤如下：

步骤1：服务器调用ACC的accessControl ABI来检索ACC进行访问控制。

步骤2：向服务器返回ACC的地址和ABI。

步骤3：服务器发送一个事务来调用ACC的accessControl ABI，该事务包含访问控制所需的信息。这个事务将被封装在一个新块中，并且直到新块被挖掘并被某个挖掘器包含在区块链中，才会执行accessControl ABI。

步骤4：在访问控制过程中，ACC调用MC中的属性，如果检测到主体的一些潜在的不当行为或者属性不匹配，ACC会根据访问主体的属性与行为得到Behavior ID，并将此结果转发给RC。

步骤5：一旦出现违规行为，RC完成了对违规行为的评判，它将把处罚返回给ACC。

步骤6：最后，在访问控制过程结束后，将访问结果同时返回给主体和对象，完成对访问主体的授权或者阻止访问。

1. 策略管理

AC在进行策略检查之前，需要通过调用addPolicy ABI进行策略添加，策略主要包括访问客体、访问行为、属性所有者、属性名称、操作符、属性值以及访问等级等几个方面，可以为访问控制提供详细的判断依据。AC可以通过getPolicy ABI遍历策略，deletePolicy ABI删除策略，updatePolicy ABI更新策略等一些列操作。

1. 策略决策

决策环节分为策略检查、行为检查以及综合决策。首先进行行为检查，主要进行频繁请求检测。然后通过操作符以及属性值的大小关系标记策略检查结果；另外需要自主定义当规则发生冲突时的策略检查结果情况。最后根据行为检查结果与策略检查结果确定behaviorID，并且给出最后的访问结果。

1. 信誉计算

可以为ACC判断主体能否进行访问控制提供先决依据，并且只有ACC可以调用RC进行信誉计算。计算数据来源ACC与MC，计算完成之后，ACC调用新的endBBN作为判断依据；MC则通过updateEndBBN ABI更新主体属性endBBN，并且只有RC可以更新该属性。通过合法行为计算信誉的正面影响，不当行为计算信誉的负面影响，将二者综合，得到访问惩罚公式，更新endBBN。

### 4.4实验结果与分析

#### 4.4.1实验环境

本文在Quorum区块链中使用智能合约实现了该访问控制系统。代码和测试在Remix，EVM版本是拜占庭，编译器版本是0.7.0，启用优化和设置200次。

工具：

* 使用Quorum Wizard命令行工具快速建立Quorum网络
* 使用Cakeshop可视化区块链和智能合约状态
* 使用Remix+Quorum Plugin for Remix的组合部署合约及合约交互

方式：

* Quorum Wizard建立网络选取Bash方式
* 建立一个4节点的私链网络进行实验，分别注册为用户设备与网关，使用cakeshop在Node1、Node2和Node3上创建新帐户，然后发送1000 ETH给新帐户。Node1 account0部署MC, Node1 account1部署RC, Node2 account1作为发送请求的用户设备，Node2 account0为account0的管理员, Node3 account1为接收请求的用户设备，Node3 account0为account1的管理员。

步骤：

1. 创建测试网络

$ quorum-wizard -v

- choose simple network

- use bash scripts

- use istanbul consensus

- 4 nodes(minimum numbers of nodes)

- Quourm 2.7.0

- Tessera 0.10.5

- use Cakeshop as chain explorer

- network name: 4-nodes-istanbul-bash

1. 部署合约，添加属性和策略

Node1 account0部署MC，Node1 account1部署RC，Node1 account1在MC中调用setRC设置RC。

Node2 account0部署ACC，Node2 account1部署ACC，Node2 account0与Node2 account1在MC中进行注册。

Node3 account0部署ACC，Node3 account1部署ACC，Node3 account0与Node3 account1在MC中进行注册。Node3 account1在ACC中调用addPolicy添加访问策略。

#### 4.4.2性能测试

性能测试分为三部分：隐私合约及交易测试，访问控制系统测试和恶意行为检测部分的测试。隐私功能测试是Quorum自带的功能，这里需要测试该功能是否已经启用。访问控制系统测试主要测试参数为完成一次访问控制的时间，期间本文通过对比信誉系统加入前后的访问控制时间来确认信誉系统的加入是否对访问控制时间有影响。恶意行为检测测试需要验证奖励、惩罚、容忍、报警四大功能。

##### 1.隐私功能测试

上文提到Quorum同时支持公共事务和私有事务。公共事务与公共以太坊一样正常工作，其中私有事务通过一个称为私有事务管理器的独立组件启用。隐私管理器组件(私有事务管理器)负责在Quorum网络上提供事务隐私。这里首先需要测试该功能是否启用。实验设置在节点2为节点3部署一个私有存储合约，使状态只允许节点3查看。当传入初始值时，如果隐私功能开启，只有节点2与节点3可以调用get函数获取该数据。

通过remix插件quorum network进行图4.6部署。

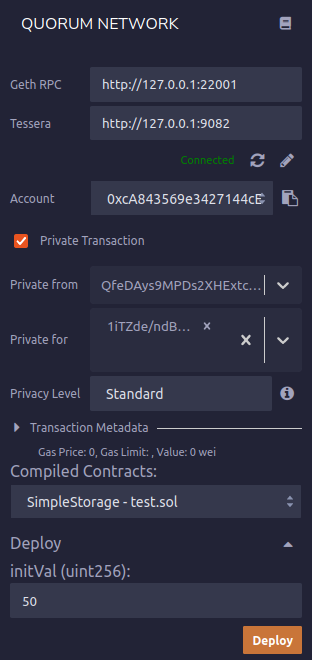


图4.6隐私功能网络配置

然后分别进入Node3和Node4的Geth console进行验证。通过调用private.get()函数进行查看，结果如图4.7和图4.8所示，从结果看Node3可以提取数据，Node4无法提取数据，说明Quorum隐私功能已经开启。

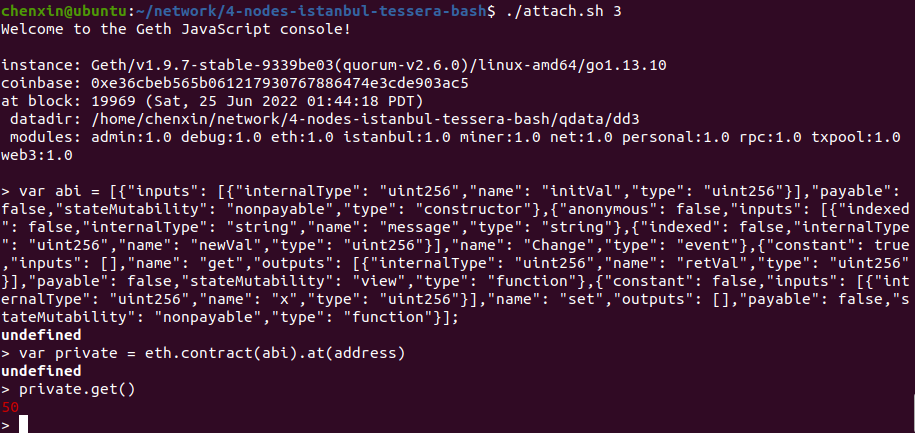


图4.7节点3状态

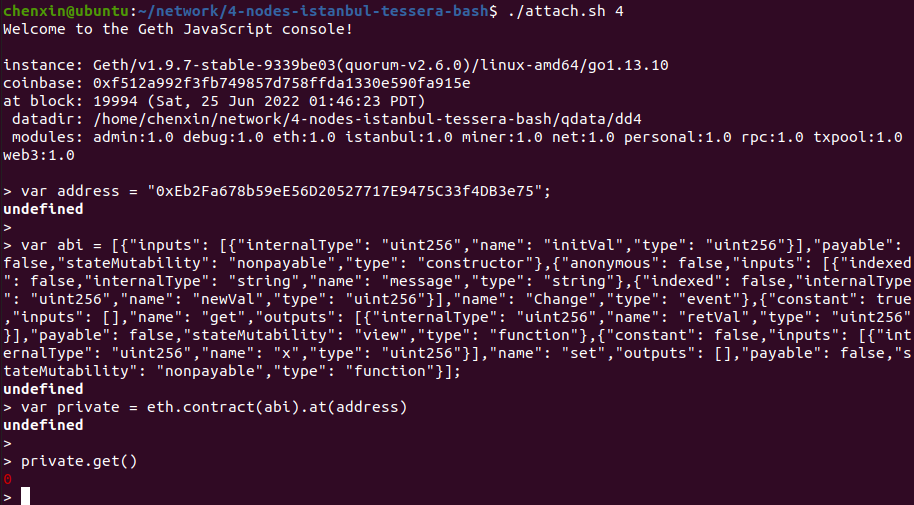


图4.8节点4状态

##### 2.访问控制时间测试

当测试一个程序或比较不同算法时，执行时间非常重要。本次测试使用Linux time命令获取执行时间。Time命令可以统计时间消耗。

1. 测试完成一次访问控制的时间

使用/user/bin/time获取执行访问控制的时间，user与sys求和，得到访问控制实际执行所花费的CPU时间。

为测试完成一次访问控制的时间，按照上文合约部署，使用node2\_account1访问node3\_account1。完成一次合法测试的算法如算法4所示：

|  |
| --- |
| **算法4：合法测试** |
| **输入: accAbi, accAddr**  **输出: result**   1. 初始化 2. const receipt←accessControl(basicInformation read)send(accountAddress gas gasPrise) 3. **if** success **then** 4. 访问成功 5. **else** 6. EVM回复 7. **end** |

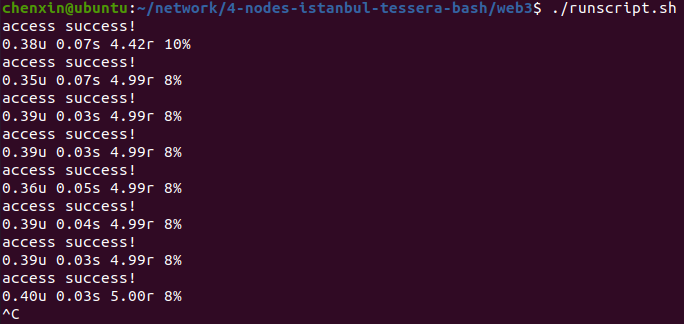


图4.9访问控制时间

如图4.9终端所示时间运行状态，第一列是用户态（user）运行时间，第二列是内核态（sys）运行时间，第三列是实际运行时间（real），最后一列是进程所获取的CPU时间比例。

1. 信誉系统的加入前后访问控制时间对比

首先删除RC，然后删除MC与ACC关于RC的相关调用，按照上文步骤进行测试，同样实现node2\_account1访问node3\_account1。这里仅进行20次访问测试。访问控制时间以及合约gas消耗如表3所示。

据表3可看出，将加入信誉系统前后的访问控制时间以及合约gas消耗简单对比，发现加入信誉系统后，合约gas消耗会随之增加。但是访问时间，经过测试发现，从平均值分析，无信誉系统时，访问时间为4.99s，加入信誉系统后，访问时间为4.97s，所以，随着信誉系统的加入，访问时间并不会有显著的增加，在功能增加的基础上可以保证原有的执行时间。

表3信誉系统加入前后访问控制时间对比

|  | 访问控制时间 | | 合约gas消耗 | |
| --- | --- | --- | --- | --- |
| ABAC+reputation | 平均值 | 4.97s | MC | 3430690 |
| 最大值 | 5.72s | ACC | 4501087 |
| 最小值 | 3.80s | RC | 2421785 |
| No\_reputation | 平均值 | 4.99s | MC | 2782804 |
| 最大值 | 5.14s | ACC | 4085323 |
| 最小值 | 4.98s |  |  |

##### 3.存储占用分析

由于不同访问控制方案的架构不同，随着平台中用户、设备及访问请求数量的增加，区块链上存储的合约大小增长也不同，对区块链造成的压力不同。

该方案与传统ABAC存储占用对比如表4所示。

表4不同方案存储占用对比

|  | 合约大小 | | |
| --- | --- | --- | --- |
| ABAC+reputation | MC | | 95881 bytes |
| ACC | | 157544 bytes |
| RC | | 65545 bytes |
| ABAC | MC | policy | 137108 bytes |
| subject | 39985 bytes |
| object | 40023 bytes |
| ACC | | 54088 bytes |

合约增长关系：

ABAC+Reputation：一个Math合约、一个MC、一个RC、n个ACC。

传统ABAC：一个subject、一个object、一个ACC、n个policy。

假设用户数量为n，两两之间会发生访问请求。合约增长表达式如下：

* ABAC+Reputation：y=153.85n+158.49
* 传统ABAC：y=133.89n+130.95

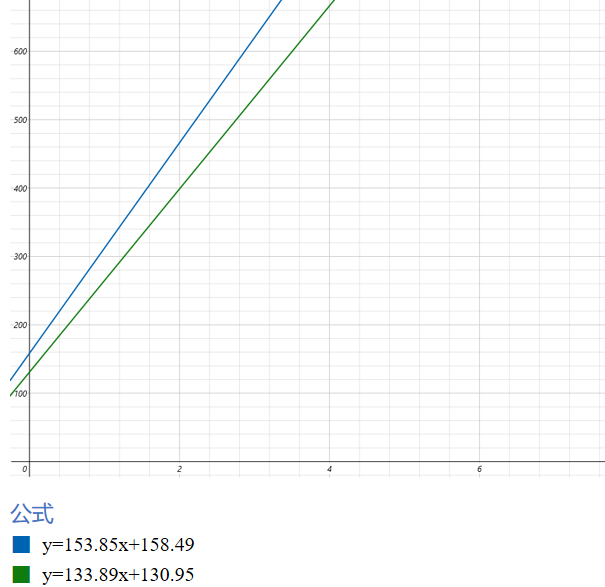


图4.10合约增长趋势

如图4.10所示，在访问控制过程中加入了信誉驱动的针对动态行为的访问控制之后，相比传统的ABAC方案，每个用户注册的特征会更加全面，访问策略也会更加具体，所以导致合约存储占用比较高，增长速率比较大。

#### 4.4.3动态行为检验测试

信誉系统的测试需要发起持续不断的合约调用，在随机时间调用随机合约函数，从而验证奖励、惩罚、容忍、报警等功能。惩罚就是阻塞时间到期之前不允许方案，是一个功能测试；奖励和容忍是一体的，由于奖励的存在，某些不太重要的的恶意行为可以被容忍而不会触发惩罚；报警是触发恶意行为时能够及时向管理员反馈，只要持续检测合约发送的事件即可。

##### 1.奖励与容忍

测试系统的奖励与容忍，我们在最短时间间隔范围外连续随机进行合法访问，访问方式如图4.11所示：

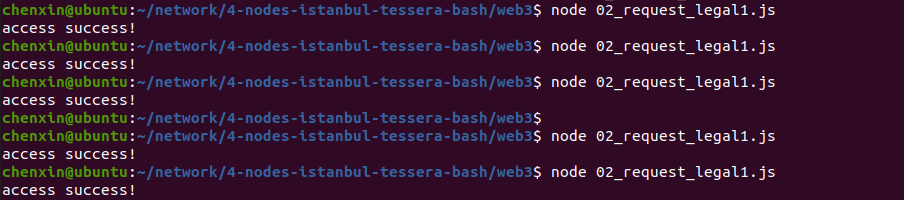


图4.11奖励与容忍测试

实验结果如图4.12所示，通过监控ACC，第一列为访问主体用户地址，第二列为客体访问控制结果，合法行为均为访问授权，非法行为为重要策略检查失败，第三列是进行的区块号，第四列是信誉值，最后是阻塞区块数。

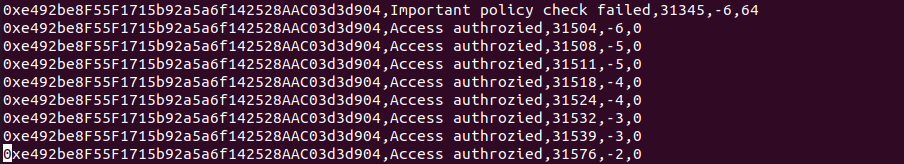


图4.12ACC监控结果

图4.13信誉值变化趋势

由图4.13可以清晰的看出，由于重要策略检查失败导致的信誉值降低，当区块号>EndBBN+阻塞区块号后，我们在时间间隔允许范围内进行连续合法访问，信誉值会随着合法访问的进行而慢慢提高，从而实现了信誉系统加入之后对有些不当行为的容忍以及对合法行为的奖励功能。因为用户都有自检功能，在未受到攻击时，能够完成自我异常行为的修正。另外如果仅仅依靠信誉值的降低，直接将异常用户剔除，会导致删除添加成本与工作复杂度。

##### 2.惩罚与报警

接下来测试的是系统的惩罚与报警，本文连续进行策略检查失败和重要策略检查失败的不当行为访问后，继续进行非法访问，测试方式如图4.14所示：

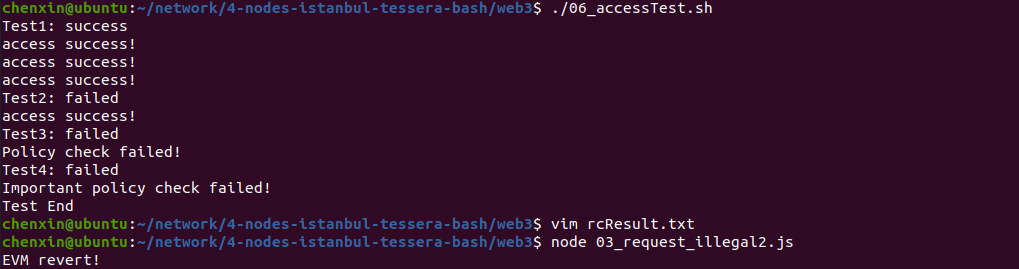


图4.14惩罚与报警测试

与上文相同，访问过程中，对ACC实行监控，第一列为访问主体地址，第二列为访问控制结果，主要有访问授权、频繁访问、策略检查失败以及重要的策略检查失败，第三列为区块号，第四列为信誉值，最后是阻塞区块数。

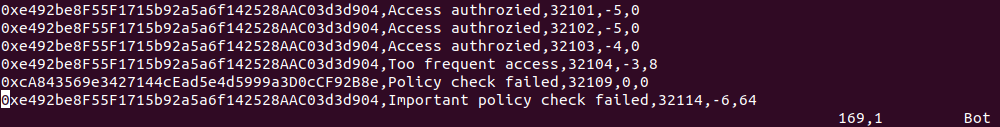


图4.15ACC监控结果

图4.16信誉值与阻塞区块数变化趋势

如图4.16所示，32101-32104仍属于访问授权的奖励过程，从32104以后，在连续非法访问之后，信誉值降低，阻塞区块数开始增加，在此期间，客体将拒绝所有来自该用户的访问请求，直至区块解封，来自其他用户的访问依旧可以进行。当用户频繁访问时，系统仍然认定访问为合法访问，但是给予了一定的区块阻塞，阻止用户的频繁访问。当出现策略检查失败，则被认定为非法请求，信誉值降低，阻塞区块号急剧增加。

### 4.5结论与展望

设计了一种基于区块链的由信誉驱动的访问控制方案，本文通过引入属性的概念，将属性进一步划分，实现细粒度的访问控制，解决访问权限难以管理的问题；引入智能合约技术，将访问控制策略编写成智能合约部署在区块链上，降低系统计算压力；引入信誉机制，解决恶意行为处理问题，实现动态访问控制。本文提出一个访问控制框架，通过对访问控制、管理合约和信誉合约等智能合约的设计，实现属性管理、策略管理和信誉计算等具体功能，进而实现了一个基于属性的静态访问权限验证和基于信誉机制的动态访问权限验证，不仅保持了基于属性的认证功能，还实现了对用户行为基于信誉值的奖惩机制。本文采用相同实验过程对加入信誉系统前后的访问控制时间进行对比，信誉系统的加入，不会造成访问时间的增加。相比传统的ABAC方案，信誉机制的加入使得每个用户注册的特征会更加全面，访问策略也会更加丰富，所以导致合约存储占用比较高，增长速率比较大。框架设计之初，由于考虑到访问控制的奖励与惩罚机制，以及删除、添加设备带来的成本问题，所以当访问者持续非法访问，本文只是持续增加该访问者的阻塞区块数已达到阻止访问的目的，而不会直接根据其信誉值的下降就将其剔出私链。

对于基于区块链的访问控制方案，针对的整个网络的数据对每个节点和用户来讲都是可见的，这就带来了一定的数据隐私安全隐患。例如，在某些应用场景下，某些敏感的数据不能以明文方式在整个网络中进行同步。为了进一步解决网络节点数据的透明性带来的安全隐私问题，我们需要一种更加细粒度的数据安全访问控制机制，即数据加密上链。通过对明文数据加密然后上链的方式，使得只拥有解密密钥的用户才能获取真正的明文数据。更进一步，我们不仅需要对数据拥有者制定的访问策略进行密文处理上链，还需要隐藏数据访问者的属性隐私，确保链上每个节点的安全性。综上所述，我们将在链上基于属性的访问控制基础上，开展进一步的研究，确保实现数据不被泄露的同时能够实现基于用户的最细粒度的数据访问控制。

### 参考文献

[1] 缪燕,孙燕,陈晓娟,宋伟.多源对象关系数据库细粒度强制访问控制机制实现方法[J].计算机应用与软件,2021,38(12):14-2155

[2] 邓集波,洪帆.基于任务的访问控制模型[J].软件学报,2003,14(1):76-82

[3] 许访,沈昌祥. 基于任务的强制访问控制模型[J]. 计算机应用研究,2004,(11):70-71+74.

[4] 尹建伟,徐争前,冯志林,陈刚,董金祥. 增强权限约束支持的基于任务访问控制模型[J]. 计算机辅助设计与图形学学报,2006,(01):143-149.

[5] Ferraiolo DF, Sandhu R, Gavrila S. Proposed NIST standard for role-based access control. ACM Trans. on Information and System Security, 2001,4(3):224−274.

[6] Mahdi Ghafoorian, Dariush Abbasinezhad-Mood, Hassan Shakeri. A Thorough Trust and Reputation Based RBAC Model for Secure Data Storage in the Cloud. IEEE Transactions on Parallel and Distributed Systems,2019,30(4):778-788.

[7] J.P.Cruz, Y. Kaji and N. Yanai. RBAC-SC: Role-Based Access Control Using Smart Contract. IEEE Access, 2018,6:12240-12251.

[8] 王静宇,冯黎晓,郑雪峰. 一种面向云计算环境的属性访问控制模型[J]. 中南大学学报(自然科学版),2015,46(06):2090-2097.

[9] 毋涛,张帆. 云计算下基于属性的访问控制方法[J]. 计算机系统应用,2016,25(02):231-234.

[10] 王于丁,杨家海. 一种基于角色和属性的云计算数据访问控制模型[J]. 清华大学学报(自然科学版),2017,57(11):1150-1158.

[11] Y. Zhu, R. Yu, D. Ma, W. Cheng-Chung Chu. Cryptographic Attribute-Based Access Control (ABAC) for Secure Decision Making of Dynamic Policy With Multiauthority Attribute Tokens. IEEE Transactions on Reliability, 2019,68(4): 1330-1346.

[12] H. Liu, D. Han, D. Li. Fabric-iot: A Blockchain-Based Access Control System in IoT. IEEE Access, 2020, 8: 18207-18218.

[13] Aftab M U, Habib M A, Mehmood N, et al. Attributed Role Based Access Control Model. Information Assurance and Cyber Security. IEEE,2016:83-89.

[14] S. Jha, S. Sural, V. Atluri, J. Vaidya. Security analysis of ABAC under an administrative model. IET Information Security.2019,13(2): 96-103.

[15] P. Wang, Y. Yue, W. Sun, and J. Liu, “An Attribute-Based Distributed Access Control for Blockchain-enabled IoT,” in 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona, Spain, Oct. 2019, pp. 1-6.

[16] Imen Riabi, Yosr Dhif, Hella Kaffel Ben Ayed, Khaled Zaatouri. “A Blockchain based access control for IoT” in 2019 15th International Wireless Communications and Mobile Computing Conference, Tangier, Morocco, June 24.2019, pp. 2086-2091.

[17] Zhang Y, Kasahara S, Shen Y, et al. “Smart contract-based access control for the internet of things,”. IEEE Internet of Things Journal, 2018., vol.6, no.2, pp. 1594-1605, April.2019

[18] 刘敖迪,杜学绘,王娜,李少卓. 基于区块链的大数据访问控制机制[J]. 软件学报,2019,30(09):2636-2654.

### 小结

本章首先分析了当前科技信息管理平台存在数据安全、用户访问等问题，明确了科技管理平台下的访问控制机制需求主要包括动态授权、细粒度定位用户以及高安全性等。根据需求，我们通过对传统访问控制的研究，提出一个基于区块链的访问控制框架，通过对访问控制合约、管理合约和信誉合约等智能合约的设计，实现属性管理、策略管理和信誉计算等具体功能，进而实现了一个基于属性的静态访问权限验证和基于信誉机制的动态访问权限验证，不仅保持了基于属性的认证功能，还实现了对用户行为基于信誉值的奖惩机制。本章在Quorum区块链中使用智能合约实现了该访问控制系统，从实验数据分析可以看出，该框架能够在不增加访问控制时间的基础上实现包括信誉驱动的动态访问控制等更丰富的访问控制功能，并且能够通过区块阻塞阻止节点的异常访问。