chiron

发布 *1.0*

2020 年 02 月 27 日

Contents

CHIRON 是一个稳定、高效、安全的区块链底层平台。

CHAPTER 1

平台介绍

1.1 背景

2008 年，中本聪划时代的提出了“一种点对点的电子现金系统”，并在白皮书公布 3 个月后提交了源代码。这份代码运行至今，无论是 BTC 本身的价值还是 BTC 的底层技术框架-区块链，都对人类社会的发展产生了前所未有的影响。2013 年，Vitalik Buterin 创办的以太坊把可编程性引入到区块链，在大幅提高了全球金融流通性的同时，让越来越多的人参与到区块链大生态共建中。

区块链技术解决方案中的共识机制、分布式账本、加密算法、智能合约、点对点通信、分布式计算架构、分布式存储、隐私保护算法、跨链协议等技术模块，可以让商业模式中的参与各方实现了地位对等和互信合作，从而推动了从“信息互联网”到“信任互联网”的时代进步，也令商业模式全面走向“分布式”成为可能。

分布式商业与此前流行的连锁加盟型商业模式及共享商业模式的最大不同之处在于，起到中间链接桥梁作用的不是人或产品、不是信息平台、而只是客观的技术本身。诚然，如果技术不开源，确实也可能演变成新的垄断。因此，发展分布式商业必须始终保持技术开源的态度，各个参与方通过开源社区进行分工合作，就将不再存在话语权集中和垄断的可能性，弱肉强食的“丛林法则”在此就不复存在。这有助于中小微企业真正成为商业价值链的主角，从而激发经济增长动力、广泛提升就业、鼓励创业和创新，实现“反垄断”的人类商业终极理想。

1.2 CHIRON

Chiron 是一个开放源代码的支持联盟链和公链的双向模式调整的区块链平台，设计用于企业环境和金融流通性，与其他流行的分布式账本或区块链平台相比，它提供了一些关键的区分功能。

1.2.1 Chiron 特点:**

- 可配置的区块链底层架构
- 支持联盟链和公链的双向模式调整
- 基于 chiron 的多链互通
- 支持交易类应用隐私保护
- 支持存证类应用隐私保护
- 基于 chiron 的不同链间应用平移
- 全栈自主知识产权
- 真正的高性能指标 N*TPS
- 开放的应用中间件社区

1.2.2 Chiron 区块链技术创新：**

- 提出一种新型的共识机制 Chiron 采用 VRF 真随机数解决去中心化问题，同时通过组内并行协作方式快速达成共识，TPS 目标 3000。另外，基于我们多年在大型分布式系统的经验，在提案、验证和出块三个环节都保证了无单点设计，进一步提高系统的性能和鲁棒性。通过严格的数学论证和工程分析，我们认为 Chiron 共识机制给出了迄今为止全球范围内不可能三角定律的最优解。
- Chiron 共识机制采用分组 VRF+BLS 的工作机制，分组 VRF 保证不能私自铸块，防御长程攻击，并对 51% 攻击的要求提升到需要控制 95% 的节点。同时 POS 权益质押与铸块收益概率线性相关，抵制了女巫攻击，并通过奖惩制度克服了无利害关系。设计应用层对通讯进行分类，需要安全信道的场合均采用 ECDH 算法对通讯信道进行加密，保证通讯的安全性。用户账户安全，我们采用零知识证明技术，确保账户的安全和隐私保护。智能合约执行安全，则在智能合约提交后，系统将对智能合约进行形式化验证，对合约的安全进行审计，确保执行的安全。
- Chiron 在设计之初就考虑了系统的效率成本，合理利用全网算力。针对 sharding 机制，Chiron 共识机制采用分组策略，从底层支持 Layer2 的分片并行计算框架。Layer2 层随机选取部分组执行分片并行计算，其他组在闲置时，定期执行 check point，对区块数据的准确性进行验证。同时 Chiron 采用分级处理的思想，对系统数据（账户，交易，智能合约等）采用冷热备份的思路，大多数节点只需存储热门的数据，提升数据存储的效率成本。
- Chiron 采用自己研发的 TVM，支持扩展行业原语作为智能合约关键词，支持多种用户友好的编程语言，如 python，go 等。并且对一些常用应用提供多个智能合约模版，用户仅需少量的设置即可调用。
- Chiron 设计了 Layer2 分片并行计算框架，支持数据分片处理，VRF 分组机制保证不同组之间可以并行处理不同任务，可将系统的吞吐量提升一个数量级。
- Chiron 将不同的设备分为轻节点和重节点两类。并根据节点对 Chiron 系统的贡献，提出了节点健康度指数。计划在第一阶段，重节点进行全量账本的存储。在第二阶段重节点实现全量账本的分布式存

储。重节点负责记账提案。轻节点仅存储部分账本以及近期高度的账户相关状态，以组协作方式对候选区块进行验证签名。即使手机作为轻节点，在有配套的 APP 情况下也可参与 Chiron 系统记账。

Chiron 试图构建一个人人可参与的高安全去中心化网络，基于这个网络可以用很低的成本开发可持续发展的商业应用。同时通过一系列数学、密码学和工程技术的引入，让这个网络在保证去中心化和高安全的同时，也是高性能和低能耗的。

CHAPTER 2

系统设计

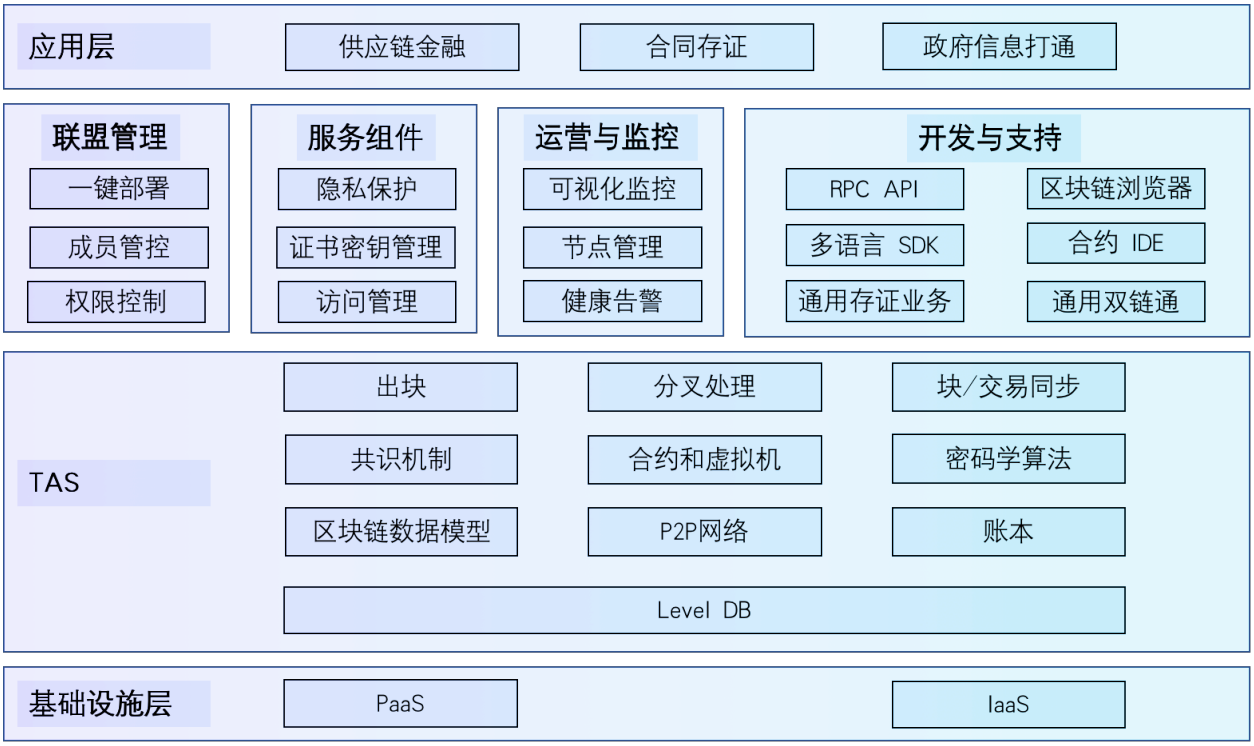
本章介绍 CHIRON 平台的设计思路，包括每个模块的结构以及实现，面向 CHIRON 平台开发者。

2.1 整体架构

整体架构上，Chiron 划分成基础层、核心层、管理层和接口层：

- **基础设施层**: 提供区块链一整套商业解决方案平台
- **区块链核心层**: 实现了区块链的核心逻辑：

  共识模块：Chiron 共识算法的核心部分，包括 VRF 随机提案，BLS 组协作验证以及一系列的密码学组件链模块：主要包括块的生成和校验，交易管理、执行和同步，分叉处理，虚拟机以及数据存储网络模块：主要包括网络的接入和退出，p2p 节点高效通讯，以及各种复杂网络中实现网络穿透等技术
- **管理层**: 实现区块链的管理功能，包括联盟管理、服务组件、应用监控、开发者工具等
- **应用层**: 面向区块链用户，提供多种协议的 RPC 接口、SDK 和交互式控制台



2.2 共识算法

2.3 虚拟机与智能合约

2.4 p2p 网络

2.5 安全控制

2.5.1 Chiron 权限文档

1.chiron 权限系统简介

chiron 权限系统是利用智能合约来控制的，在本权限系统中，设计上共采用 7 个合约，他们分别是：

`PermissionsUpgradable` 合约：只能通过 guardian 用户来调用的合约，一般在初始化的时候来绑定各个合约地址或者当 `PermissionImplementation` 合约逻辑更改来升级合约时候使用。

`PermissionsInterface` 合约：接口合约，用户对权限的操作需调用该合约中封装的接口来调用，该合约会调用 `PermissionsImplementation` 合约来进行进一步调用。

`PermissionsImplementation` 合约：通过上面的 `PermissionsInterface` 合约来进行统一调用的，该合约进而对底层的合约进行调用，考虑到以后存在由于逻辑变动而要升级合约的情况，该合约是可以被 `guardian`

重新绑定以进行升级操作。

`OrgMgr` 合约：对组织进行存储的合约，包含组织列表，组织状态等等。`PermissionsImplementation` 合约对组织的操作最终都会回到该合约中操作。通过 `PermissionsImplementation` 合约调用。

`VoteMgr` 合约：进行投票管理的合约，该合约中存储投票者和各投票事项。一般在联盟层面上的操作都需要联盟管理员进行申请和投票两步操作才可以。通过 `PermissionsImplementation` 合约调用。

`AccountMgr` 合约：对账户进行管理的合约。通过 `PermissionsImplementation` 合约调用。

`NodeMgr` 合约：对节点进行管理的合约，可以指定某个节点是否有挖矿权限。通过 `PermissionsImplementation` 合约调用。

2.chiron permission APIs

↓↓↓↓↓ 以下基于联盟内的操作都需要通过联盟管理员操作，且需要两步完成，即申请 + 批准 ↓↓↓↓↓

add_org

用来在联盟中添加一个新组织时候使用。该方法仅可以被联盟管理员调用

Parameters

`_org_id`: 新添加的组织 id `_account`: 新添加组织中的账户 id，该账户会被自动指定为该组织的管理员 `_node_id`: 新添加组织中的节点 id

---

approve_org

用来在联盟中投票通过一个新组织时候使用。该方法仅可以被联盟管理员调用，与前面的 `add_org` 方法是相关联的操作，前面通过 `add_org` 添加的组织需要联盟管理员调用此方法进行投票投票且超半数方可成功加入。

Parameters

`_org_id`: 需要 approve 操作的组织 id `_account`: 需要 approve 操作组织中的账户 id，该账户会被自动指定为该组织的管理员 `_node_id`: 需要 approve 操作组织中的节点 id

---

update_org_status

由联盟管理员执行，当需要更改非联盟管理员组织的状态的时候使用。

Parameters

_org_id: 需要更改状态的组织 id _action: 操作符，1 表示暂停组织，2 表示将组织从已暂停的状态中恢复为激活状态

---

approve_org_status

由联盟管理员帐户执行，并用于批准组织状态更改建议。从网络管理员那里获得多数批准后，组织状态就会更新。

Parameters

_org_id: 需要更改状态的组织 id _action: 操作符，1 表示暂停组织，2 表示将组织从已暂停的状态中恢复为激活状态

---

assign_alliance_admin

由联盟管理员帐户执行，用于向联盟组织中添加一个新的联盟管理员账户时使用

Parameters

_org_id: 联盟管理员组织 id _account: 需要指派为联盟管理员的账户 id

---

approve_alliance_admin

由联盟管理员帐户执行，用于批准向联盟管理员组织中新加管理员的建议。获得多数联盟管理员批准后，组织中便会增加一个有效的联盟管理员账户。

Parameters

_org_id: 联盟管理员组织 id _account: 需要批准为联盟管理员的账户 id

---

add_miner_node

由联盟管理员帐户执行，用于向联盟中新添加一个挖矿节点时使用

Parameters

_node_id: 该节点的 id _org_id: 需要添加到目的组织的 id _miner_role: 表示矿工类型。1 表示提案矿工，2 表示验证矿工 _vrf_pk:vrf_pk _bls_pk:bls_pk _weight: 表示矿工权重

---

approve_miner_node

由联盟管理员帐户执行，用于批准向联盟中新添加一个挖矿节点时使用，同样需要得到多数的联盟管理员批准后方可生效。

Parameters

_node_id: 要批准的该节点的 id _org_id: 要批准的需要添加到目的组织的 id _miner_role: 要批准的矿工类型。1 表示提案矿工，2 表示验证矿工 _vrf_pk: 要批准的 vrf_pk _bls_pk: 要批准的 bls_pk _weight: 要批准的矿工权重

---

assign_node_to_miner

由联盟管理员帐户执行，用于将联盟中某个已存在的普通同步节点指定为矿工节点

Parameters

_node_id: 该节点的 id _org_id: 该节点所存在的组织的 id _miner_role: 要分配的矿工类型。1 表示提案矿工，2 表示验证矿工 _vrf_pk:vrf_pk _bls_pk:bls_pk _weight: 矿工权重

---

approve_node_to_miner

由联盟管理员帐户执行，用于批准将联盟中某个已存在的普通同步节点指定为矿工节点的请求，需要得到多数的联盟管理员批准后方可生效。

Parameters

_node_id: 要批准的该节点的 id _org_id: 要批准的需要添加到目的组织的 id _miner_role: 要批准的矿工类型。1 表示提案矿工，2 表示验证矿工 _vrf_pk: 要批准的 vrf_pk _bls_pk: 要批准的 bls_pk _weight: 要批准的矿工权重

---

update_miner_status

由联盟管理员帐户执行，用于更改节点的矿工状态时使用

Parameters

_node_id: 要批准的该节点的 id _org_id: 要批准的需要添加到目的组织的 id _action: 操作符，1 表示暂停矿工，2 表示矿工从已暂停的状态中恢复为激活状态

---

approve_miner_status

由联盟管理员帐户执行，用于批准更改节点的矿工状态请求时使用。当得到多数的联盟管理员批准后方可生效。

Parameters

_node_id: 要批准的该节点的 id _org_id: 要批准的需要添加到目的组织的 id _action: 操作符，1 表示暂停矿工，2 表示将矿工从已暂停的状态中恢复为激活状态

↓↓↓↓↓ 以下基于组织内的操作都需要通过组织管理员操作 ↓↓↓↓↓

add_account

由组织管理员帐户执行，用于向组织中新添加一个账户时使用。

Parameters

_account: 新添加的账户 id _org_id: 新添加的账户所从属的组织，该组织必须存在且为 approved 状态 _access: 读写权限，分为四个等级。用户可指定 0,1,2 三个等级，高级别兼具有低级别的权限：0:ACCESS_READONLY，只读权限 1:ACCESS_TRANSACT，可以发送交易 2:ACCESS_CONTRACT_DEPLOY，可以部署合约 3:ACCESS_FULL_ACCESS，全访问权限 _is_admin:bool 值，表示该账户是否为管理员 **注意：如果 _is_admin 设置为 true 时，_access 的值不能为 0 只读状态**

---

update_account_status

由组织管理员帐户执行，用于更改组织中某个账户的状态时使用。

Parameters

_account: 要更改的账户 id _org_id: 要更改的账户所从属的组织 _action: 操作符，1 表示暂停使用该账户，2 表示将已暂停的账户恢复为激活状态

---

update_account_access

由组织管理员帐户执行，用于更改组织中某个账户的读写权限时使用。

Parameters

_account: 要更改的账户 id _org_id: 要更改的账户所从属的组织 _access: 想要更改的目的读写权限，分为四个等级。用户可指定 0,1,2 三个等级，高级别兼具有低级别的权限：0:ACCESS_READONLY，只读权限 1:ACCESS_TRANSACT，可以发送交易 2:ACCESS_CONTRACT_DEPLOY，可以部署合约 3:ACCESS_FULL_ACCESS，全访问权限

---

add_node

由组织管理员帐户执行，用于向组织中新增一个同步节点时使用。

Parameters

_node_id: 新增的节点 id _org_id: 新增节点所从属的组织 id

---

update_node_status

由组织管理员帐户执行，用于更改组织中某个节点的状态时使用

Parameters

_node_id: 新增的节点 id _org_id: 新增节点所从属的组织 id _action: 操作符，1 表示暂停该节点，2 表示将已暂停的节点恢复激活状态

账户权限说明：

| 权限 | 值 | 说明 | | —| —| —| | ACCESS READONLY | 0 | 只读权限 | | ACCESS TRANSACT | 1 | 发送交易权限，同时具备 0 权限 | | ACCESS CONTRACT DEPLOY | 2 | 部署合约权限，同时具备 0，1 权限 | | ACCESS FULL ACCESS | 3 | 全访问权限，为联盟管理员特有，同时具备 0，1，2 权限 |

节点矿工类型说明：

| 矿工类型 | 值 | | —| —| | PROPOSAL MINER | 1 | | VERIFY MINER | 2 |

节点状态说明：

| 状态描述 | 值 | | —| —| | NOT IN LIST | 0 | | PENDING APPROVAL | 1 | | ACTIVE | 2 | | SUSPENDED | 3 |

矿工状态说明：

| 状态描述 | 值 | | —| —| | MINER NO ACTION | 0 | | MINER PENDING ACTIVATE | 1 | | MINER ACTIVATED | 2 | | MINER PENDING ABORT | 3 | | MINER ABORTED | 4 | | MINER ABORTED REVOKE | 5 |

组织状态说明：

| 状态描述 | 值 | | —| —| | NOT IN LIST | 0 | | PROPOSED | 1 | | APPROVED | 2 | | PENDING SUSPENSION | 3 | | SUSPENDED | 4 | | PENDING SUSPENSION REVOKE | 5 |

账户状态说明：

| 状态描述 | 值 | | —| —| | NOT IN LIST | 0 | | PENDING APPROVAL | 1 | | ACTIVE | 2 | | SUSPENDED | 3 |

投票类型：

| 投票类型 | 值 | | —| —| | VOTE OP ADD ACTIVITY ORG | 1 | | VOTE OP SUSPEND ORG | 2 | | VOTE OP REVOKE SUSPEND ORG | 3 | | VOTE OP ASSIGN ALLIANCE ADMIN | 4 | | VOTE OP REMOVE ALLIANCE ADMIN | 5 | | VOTE OP ADD MINER NODE | 6 | | VOTE OP ASSIGN NODE TO MINER | 7 | | VOTE OP UPDATE MINER STATUS | 8 |

2.6 存储模块

2.7 数据结构

2.8 RPC

CHAPTER 3

安装部署

本章介绍 CHIRON 所需的必要安装和配置。以及安装步骤

CHAPTER 4

使用手册

本章提供了 CHIRON 平台的使用手册，使用手册介绍 CHIRON 平台各种功能使用方式。

CHAPTER 5

JSON-RPC API

下列接口的示例中采用curl命令，curl 是一个利用 url 语法在命令行下运行的数据传输工具，通过 curl 命令发送 http post 请求，可以访问 CHIRON 的 JSON RPC 接口。curl 命令的 url 地址设置为节点配置文件 `[rpc]` 部分的 `[listen_ip]` 和 `[jsonrpc listen port]` 端口。为了格式化 json，使用jq工具进行格式化显示。

CHAPTER 6

区块链浏览器

6.1 一、简介

描述 1

6.2 二、区块链浏览器搭建

搭建

6.3 三、功能简介

简介

CHAPTER 7

SDK

CHIRON 向外部暴露了接口，外部业务程序能够通过 CHIRON 提供的 SDK 来调用这些接口。开发者只需要根据自身业务程序的要求，选择相应语言的 SDK，用 SDK 提供的 API 进行编程，即可实现对区块链的操作。

CHAPTER 8

常见问题解答

##example1

CHAPTER 9

社区

introduction

##example1