软件学报 ISSN 1000-9825, CODEN RUXUEW
2025,36(10):4612−4627 [doi: 10.13328/j.cnki.jos.007311] [CSTR: 32375.14.jos.007311]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

面向联盟链的智能合约行为可信验证机制*

张建标[1,2], 康 双[1,2], 张兆乾[3], 刘德田[1,2]

[1](北京工业大学 计算机学院, 北京 100124)

[2](可信计算北京市重点实验室, 北京 100124)

[3](中国长江三峡集团有限公司 科学技术研究院, 北京 100038)

通信作者: 张建标, E-mail: zjb@bjut.edu.cn

摘 要: 针对联盟链系统中恶意攻击者利用智能合约中的安全漏洞带来的行为不可信问题, 提出一种面向联盟链的智能合约行为可信验证机制对合约行为完整性进行可信验证. 首先以系统调用作为最小行为单元, 以基于系统调用的行为序列描述历史行为状态, 然后在确保合约代码发布和执行环境可信的前提下, 在合约运行时根据预期行为规则进行可信验证, 最后对该机制进行了理论分析, 并在 Hyperledger Fabric 环境下进行实验评估. 结果表明, 该方法能够有效实现对智能合约行为的可信验证, 能够保障智能合约生命周期内的行为可信.

关键词: 联盟链; 智能合约; 可信验证; 主动度量

中图法分类号: TP311

中文引用格式: 张建标, 康双, 张兆乾, 刘德田. 面向联盟链的智能合约行为可信验证机制. 软件学报, 2025, 36(10): 4612–4627. http://www.jos.org.cn/1000-9825/7311.htm

英文引用格式: Zhang JB, Kang S, Zhang ZQ, Liu DT. Trusted Verification Mechanism of Smart Contract Behaviour for Consortium Blockchain. Ruan Jian Xue Bao/Journal of Software, 2025, 36(10): 4612–4627 (in Chinese). http://www.jos.org.cn/1000-9825/7311.htm

Trusted Verification Mechanism of Smart Contract Behaviour for Consortium Blockchain

ZHANG Jian-Biao[1,2], KANG Shuang[1,2], ZHANG Zhao-Qian[3], LIU De-Tian[1,2]

[1](College of Computer Science, Beijing University of Technology, Beijing 100124, China)

[2](Beijing Key Laboratory of Trusted Computing, Beijing 100124, China)

[3](Institute of Science and Technology, China Three Gorges Corporation, Beijing 100038, China)

Abstract: To address the issue of untrustworthy behaviors resulting from malicious attackers exploiting security vulnerabilities within smart contracts in the consortium blockchain system, this study introduces a trusted verification mechanism of smart contract behavior for consortium blockchain to conduct trusted verification for contract behavior integrity. Firstly, the proposed approach takes the system call as the smallest behavior unit and describes the historical behavioral state with the behavior sequence based on system calls. Subsequently, on the premise of ensuring the trustworthiness of contract code release and the execution environment, it performs trusted verification according to predefined behavioral rules during contract execution. Finally, a theoretical analysis of this mechanism is carried out, and an experimental evaluation is conducted in the Hyperledger Fabric environment. Results demonstrate that the proposed method can effectively achieve the trusted verification of smart contract behavior and ensure the credibility of behavior within the life cycle of smart contracts.

Key words: consortium blockchain; smart contract; trusted verification; active measurement

区块链是一种去中心化的分布式数字账本, 具有不可篡改、透明、可追溯的特性[1]. 智能合约作为部署在区块链上能够根据条件自动执行的计算机代码[2], 扩展了区块链的功能. 特别是在联盟链领域, 智能合约可以支持金融、股票、电网、医疗等多种业务场景[3−6]. 联盟链是介于公有链和私有链之间的一种折中方案, 具有弱中心化、

* 基金项目: 北京市自然科学基金 (M21039)
收稿时间: 2023-10-17; 修改时间: 2024-04-01; 采用时间: 2024-10-28; jos 在线出版时间: 2025-04-30
CNKI 网络首发时间: 2025-05-06

交易效率高、访问控制等特点[7]. 相比公有链, 联盟链更适用于重要信息系统, 具有更加广泛的应用场景.

近年来, 智能合约一直受到安全事件的困扰. 智能合约本质上是一种计算机代码, 不可避免存在代码漏洞, 这也是目前针对合约进行攻击的常用手段. 例如, 2016 年的 DAO 事件导致价值约 6 000 万美元的以太币损失[8]. 2018 年, 攻击者针对 Spankchain 智能合约发起重入攻击, 并盗走 165 个 ETH[9]. 2022 年, 将近 8 000 万美元的被盗资产令 Fei Protocol 成为有史以来规模最大的重入攻击受害者[10]. 这些事件都暴露了智能合约存在着严重的安全漏洞, 威胁着区块链系统的稳定性. 因此, 对于具有严格安全需求的联盟链应用场景, 例如情报机构、金融机构等, 不能简单地要求没有漏洞的合约, 而是应要求合约的行为符合预期.

对于智能合约来说, 其行为主要体现在对账本的读写操作上. 本文采用主动免疫可信计算的概念, 在计算运算的同时进行安全防护, 使计算结果符合预期[11], 因此, 智能合约行为可信是指合约在执行过程中没有发生任何违反安全策略、未经授权或恶意的操作, 能够达到预期的效果. 智能合约行为不同于一般的软件行为, 智能合约行为是不可逆的, 一旦合约被部署到区块链上, 就无法修改或删除. 智能合约行为是公开透明的, 任何人都可以查看合约的源代码. 智能合约行为是受限制的, 合约不能直接访问外部数据或服务, 只能通过预定义的接口与外部环境交互. 在联盟链中, 智能合约行为可信面临以下几种威胁: (1) 合约漏洞威胁: 合约代码中存在逻辑错误或安全漏洞, 导致合约被攻击者利用; (2) 合约篡改威胁: 合约存储在区块链, 运行在本地沙箱环境中, 从区块链到沙箱的过程发生合约的篡改, 使得合约的执行结果发生错误. 例如在金融投资场景中[12], 攻击者利用合约中的回退函数漏洞抛出异常, 导致资金以不符合系统预期的方式发生逆转, 金融机构不得不支付大量赎金以请求攻击者撤销抛出操作. 因此, 保证联盟链系统中智能合约的行为可信至关重要.

然而, 现有的智能合约漏洞检测技术并不完全适用于具有较高安全需求的联盟链应用场景. 一方面, 大部分研究方案[13,14]都是对合约代码漏洞进行检测, 主要针对智能合约安全性进行研究, 而对智能合约可信性的研究较少. 另一方面, 目前智能合约可信性研究方案主要集中在利用可信执行环境保障合约数据的隐私, 并未考虑合约行为可信. 此外, 目前的研究方案仅在合约部署前发现漏洞, 并未考虑合约运行时的安全问题.

可信计算为智能合约可信研究提供了新思路, 安全可信是国家网络安全法律、战略和等级保护制度的要求. 等保 2.0[15]中明确指出针对重要信息系统及应用进行动态可信验证的要求, 强化了网络安全等级保护一级到四级的可信要求, 以确保重要信息系统的安全. 因此, 基于区块链技术构建的重要信息系统也需要满足区块链应用程序动态可信验证的要求, 而智能合约作为扩展区块链功能的主要技术, 智能合约行为的不可信必将导致区块链系统的不可信, 对智能合约行为进行可信验证更是重中之重.

针对上述问题和挑战, 本文基于可信计算技术提出一种面向联盟链的智能合约行为可信验证机制, 在合约所有执行环节进行动态可信验证. 基于控制流图提取合约预期行为规则, 作为行为可信验证的依据. 在合约执行过程中对行为进行主动控制, 基于预期行为规则对其进行动态可信验证, 保障智能合约的可信运行. 此外, 本文分析了合约行为的控制流, 选择 Hyperledger Fabric 环境下具有代表性的合约验证了该机制的可行性. 实验结果表明, 本文提出的智能合约行为可信验证机制能够在区块链系统中建立起主动免疫机制, 确保合约的行为按照预期执行.

本文的主要贡献如下.

(1) 从行为角度分析智能合约, 提出智能合约行为可信的定义, 以系统调用作为最小行为单元, 用基于系统调用的行为序列描述合约行为.

(2) 提出基于主动免疫可信计算的智能合约行为可信验证机制, 实时监控合约运行时的系统调用和状态变化, 根据预期行为规则进行动态可信判定, 实现对合约运行时的可信度量与主动防护, 提升智能合约的可信性.

(3) 在 Hyperledger Fabric 环境下进行实验验证, 实验结果表明, 我们提出的智能合约行为可信验证机制能够有效实现对合约行为的验证, 确保智能合约行为是符合预期的.

本文第 1 节介绍并总结相关工作. 第 2 节介绍背景和基础知识. 第 3 节介绍我们提出的智能合约行为可信验证机制的设计与实现细节. 第 4 节分析该机制的安全性. 第 5 节通过实验评估该机制的效果. 第 6 节总结全文并展望未来工作.

1 相关工作

目前针对智能合约行为的研究包括代码检测、机器学习、可信计算技术等. 代码检测主要包括符号执行、形式化验证、模糊测试等. Zheng 等人[16]提出了一个用于智能合约并行分叉符号执行的通用框架 Park, 其主要思想是在符号执行过程中使用多个进程, 利用多个 CPU 核心来提高效率. Hu 等人[17]提出智能合约工程的概念, 通过一致性测试方法用形式化语言描述行为测试集, 观察测试用例, 并记录被测试合约的行为响应. Kalra 等人[18]提出 ZEUS 框架, 以结合抽象解释和符号模型检验的方法对智能合约进行自动的形式化验证, 可准确地根据字节码的执行语义推理出合约行为. Lv 等人[19]提出了一种针对联盟链的静态分析方法, 该方法主要包括 3 个模块: 合约静态分析、检测执行和生成可视化报告, 该方法基于抽象语法树、函数依赖等方法提取合约已知风险的静态结构特征形成特征库, 检测执行模块将静态分析得到的合约静态结构特征与特征库进行匹配来确定潜在风险的类型和位置, 最后生成可视化报告. Ding 等人[20]提出了一种基于模糊技术的 Hyperledger Fabric 智能合约检测方法 HFContractFuzzer, 该方法使用 MockStub 类在本地实现对智能合约接口的调用, 将智能合约的单元测试用例作为初始数据库, 然后再大量随机数据输入到智能合约中, 其结果与数据库数据进行比较以检测智能合约中的逻辑漏洞. 代码检测虽然可以有效地避免合约漏洞所带来的风险, 但是代码检测仅能够在合约部署前对合约进行验证, 未考虑合约运行时的状态.

Hu 等人[21]提出了一个基于机器学习中 LSTM 模型的智能合约漏洞检测方案, 该方案从合约交易行为中提取出行为模式, 并对每个类别的行为进行特征提取和异常检测. Liu 等人[22]提出了一种结合图神经网络和专家知识的时间消息传播网络. 从归一化图中提取图特征, 并将图特征与设计的专家模式相结合以产生最终的检测系统, 能够提高对智能合约漏洞检测的准确性. 这些方法可以有效地识别已知类型的漏洞, 例如整数溢出、重入攻击、异常处理等, 从而提高智能合约的安全性, 但无法检测到合约是否被篡改, 无法满足联盟链应用的需要.

一些研究方案使用可信计算技术来确保合约运行的可信. Chen 等人[23]提出 TeeSwap 将智能合约状态加载到可信执行环境 TEE (trusted execution environment) 中来保证智能合约运行过程中的隐私性. Cheng 等人[24]提出 Ekiden 系统通过将智能合约状态加载到 TEE 中, 从而保证交易执行前的数据完整性. Yan 等人[25]提出一种通过 TEE 支持链上机密性的系统设计方案 CONFIDE 保证数据的机密性和完整性. Brandenburger 等人[26]提出基于 SGX (software guard extensions) 的智能合约执行架构和原型, 该架构可以适应 Hyperledger Fabric 模式, 并将每个应用封装在安全隔离环境中. Keleket 等人[27]提出一种基于 SGX 的轻量 Fabric 链码可信执行环境构建方法 E-Fabric 架构, E-Fabric 在 Enclave 中执行链码并安全地操作用户数据, 以保证链码运行时的隐私安全. 这些方法利用硬件或软件提供的安全隔离机制, 来防止智能合约的数据或代码被篡改或泄露, 从而提高智能合约的保密性和完整性. 这些工作侧重于通过可信硬件为智能合约创建隔离的执行环境, 并未对合约行为的可信进行关注, 仅提升合约的安全性而忽略可信性是局限的, 也并不满足联盟链中高安全需求应用场景的需要.

对于合约行为分析, 不仅需要关注合约代码发布的完整性, 还要能够覆盖合约的运行时状态和行为, 动态度量为智能合约的行为可信提供了一种新的思路.

在行为动态度量方面, 已有研究的主要技术包括系统调用短序列模型、静态分析与动态分析. 研究表明[28], 程序在正常运行时所产生的系统调用与程序运行异常时产生的系统调用在时序上有明显的差异, 因此系统调用非常适合描述程序行为. Forrest 等人[29]将 UNIX 系统中进程的正常行为定义为进程所产生系统调用的短序列. 静态分析是一种通过反编译对二进制源码进行语法分析从而提取函数调用或系统调用序列, 根据调用关系构造控制流图并以此构建检测模型的方式. Onwuzurike 等人[30]对应用程序源代码进行静态分析获取 API 调用序列作为预期行为序列, 并通过实际行为与预期行为匹配评估行为可信性. 动态分析是在沙箱等环境中执行程序, 监控运行时的行为, 依据历史执行结果等信息对软件预期系统调用序列进行学习, 从而构建正常行为模型. Mishra 等人[31]提出一种动态分析方法 VMAnalyzer 提取所有系统调用的有序序列来检测恶意系统调用序列的行为.

基于行为的动态度量符合国内外主流可信计算组织可信定义的本质, 能够有效反映系统运行过程的可信, 因此通过动态度量研究智能合约的行为可信是一种可行的方式.

2 基础知识

2.1 区块链及智能合约

区块链是去中心化的数字交易账本, 由区块链网络中的节点在共识协议的约束下共同维护. 共识协议能够保证区块链网络中诚实节点在恶意节点干扰下也能达成共识, 共识的过程是各节点验证及更新账本的过程, 共识的结果是系统对外提供一份统一的账本. 区块链中的交易被保存在一个不断增长的有序“块”列表中, 每个块还包括状态元数据、创建时间戳、事务的 Merkle 哈希值、链中前一个块的哈希值以及智能合约代码和数据[32], 区块链账本数据结构如图 1 所示, 一旦交易发布, 便不可篡改.

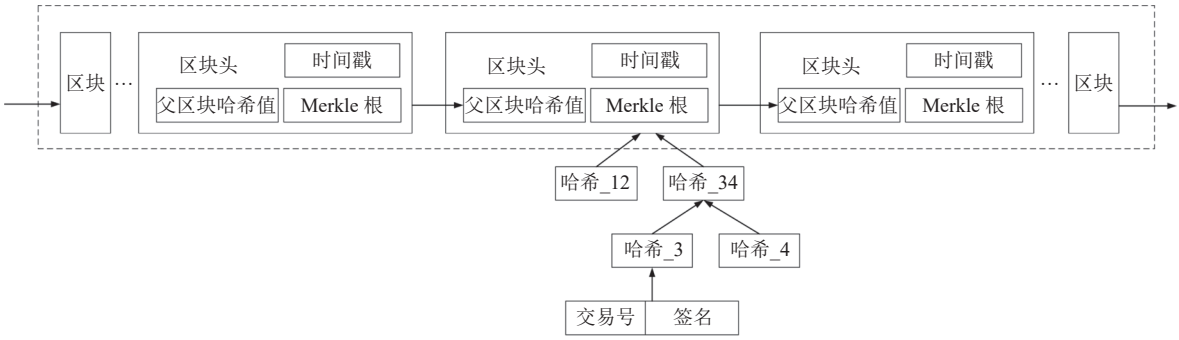


图 1 区块链账本数据结构

区块链系统根据其开放程度和参与主体的不同, 可分为公有链、联盟链及私有链这 3 种形态. 其中, 公有链对所有人开放, 允许任何人加入网络并参与区块验证过程, 以达成共识. 私有链完全由某个组织控制, 该组织拥有绝对的权威, 节点完全受组织监管, 监管组织可以快速发现节点篡改数据等错误源. 联盟链是指由多个机构管理的区块链, 只有一组有限的已批准的参与者才有权验证交易, 这种受限制的模型提供了更好的隐私性、可扩展性和细粒度访问控制. 因此, 大多数金融、医疗机构等都遵循这种模式, 目前流行的联盟链区块链平台有 Hyperledger Fabric, Quorum, Cordite 等.

2.1.1 智能合约生命周期

1994 年, Szabo[33]将智能合约定义为执行合同条款的计算机化交易协议. 随着区块链技术的发展, 智能合约被应用在区块链中. 智能合约是具有相关方之间协议条款的自动执行合同, 合约以程序代码的形式编写并存储在区块链上, 经过区块链的共识机制正确执行, 并由节点验证后, 记录在区块链中. Hyperledger Fabric[34]作为联盟链的代表, 其智能合约又称为链码 (chaincode), 可由 JavaScript 或 Go 语言编写, 其智能合约的生命周期主要包括合约的创建、发布和调用.

(1) 合约的创建是指由一个区块链系统中的节点在本地的智能合约编辑器中编写源代码, 创建一个智能合约创建交易并发送到区块链系统中的其他节点进行交易验证.

(2) 合约的发布是指由一个区块链系统中的共识节点收集到节点发送的智能合约创建交易后, 验证交易并在区块链上部署该智能合约.

(3) 合约的调用是指一个节点发起智能合约调用交易来调用智能合约, 共识节点在收集到智能合约调用交易后, 验证该交易; 从区块链账本中获取合约源代码, 在本地合约执行环境 (包括虚拟机、Docker 容器等) 中运行智能合约, 得到的结果将写入到智能合约调用交易中; 该交易最终会与其他交易打包进一个区块中, 若区块被区块链网络接受则智能合约的调用生效.

2.1.2 智能合约不可信行为

智能合约不可信行为是指合约在运行过程中产生的与预期不符的行为, 理论上合约的程序逻辑设计是正确的, 能够实现智能合约的功能. 然而合约编码过程中程序逻辑控制可能导致合约缺陷, 攻击者可能通过特定的输入

实现恶意的程序执行, 导致智能合约执行出现问题. 目前的研究方案缺乏对智能合约可信性的分析, 本文从访问控制、代码漏洞和资源完整性这 3 个方面对智能合约不可信行为进行分析.

(1) 访问控制: 合约缺乏权限验证可导致受保护的数据或功能被恶意调用. 例如没有背书权限的节点对交易进行恶意背书使得系统中的数据变得不一致, 未经授权的节点试图绕过访问控制策略非法访问区块链资源.

(2) 代码漏洞: 恶意攻击者利用代码中的漏洞发起攻击. 当合约调用另一个合约时, 如果合约的调用被攻击者窃取, 则该合约将被迫执行其他恶意代码, 包括对自身的回调, 导致相同的合约代码会像递归函数调用一样被重复执行. 因此带有恶意调用函数的合约, 可能会对受害合约发起递归调用, 绕过检查以重复获得收益.

(3) 资源完整性: 攻击者破坏智能合约运行过程中的资源完整性. 合约运行过程首先要保障合约代码的完整性. 合约存储在区块链上, 虽然合约上链后是无法删除或修改的, 但合约从链上拿到本地的过程是有可能被篡改的. 智能合约一般运行在隔离的沙箱中, 一旦在运行环境中存在自身安全缺陷或控制机制不完善等问题, 攻击者可通过部署恶意合约代码, 扰乱合约业务逻辑. 因此需要考虑合约运行环境的完整性. 此外, 智能合约可能会调用区块链内部和外部数据资源, 例如区块链文件、外部库的调用等. 因此需要保证其完整性. 如果向节点返回不可信的外部资源, 则可能会对区块链系统造成威胁.

2.2 可信计算

2.2.1 可信定义

可信计算是一种以密码学为基础, 以可信芯片为信任源头的增强计算机系统可信性的综合性信息安全技术[35]. 通过在计算系统平台中构建一个作为信任起点的信任根, 并通过信任链将信任关系从底层硬件平台逐步扩展至上层应用, 从而确保计算资源的完整性和行为的预期性. 目前, 关于可信尚未形成统一定义, 不同组织解释方式有所不同, 其中较具代表性的为以下几种.

(1) 可信计算组织 TCG (trusted computing group) 定义可信为: 如果实体的行为总是以预期的方式, 实现预期的目标, 则该实体可信[36].

(2) 沈昌祥院士[11]结合人类免疫系统的理念提出了主动免疫可信计算: 在计算、运算的同时进行安全防护, 以密码为基因实施身份识别、状态度量、保密存储等功能, 及时识别“自己”和“非己”成分, 从而破坏并排斥进入机体的有害物质, 相当于为网络信息系统培育了免疫能力, 使过程和操作行为在任意条件下的计算结果总是符合预期, 开启了我国可信 3.0 防御与运算并行的网络安全主动防御时代.

2.2.2 行为动态度量

可信度量是一种通过基准值验证数据完整性或评估实体行为和预期描述的信任量化方法[37], 主要包括静态度量和动态度量. 其中静态度量主要对数据完整性加载时进行可信校验, 而动态度量更侧重程序运行过程是否符合预期. 由于程序运行过程中的复杂性, 静态度量已不能满足系统运行时的安全需求, 不符合行为符合预期的可信本质. 动态度量是一种对信任持续、动态的量化手段, 动态度量技术涉及的 4 个要素分别是主体、客体、操作和环境. 动态度量在系统运行过程中, 通过对系统相关组件或数据的完整性、实体行为等多类对象进行持续的综合可信评估. 行为动态度量通过行为特征提取构建预期行为集, 通过行为监控获取实际行为集, 通过比较实际行为集与预期行为集的一致性对行为进行可信评估.

3 智能合约行为可信验证机制

3.1 智能合约行为基本定义

本文提出一种针对合约行为的可信验证机制来保障合约行为符合预期, 参考软件行为学中的行为定义[38], 研究如何构建合约的实时度量和防护方法. 智能合约行为基本定义, 包含以下元素.

(1) 区块链系统状态集 $S=(s_0,s_1,...,s_n)$, 区块链原始状态为 s_0;

(2) 单个行为单元集 A, 行为单元 a 是构成某个行为的原子操作即系统调用;

(3) 单个合约的行为集 B, 其中行为 α 由所有行为单元 a 连接而成;

(4) 预期行为规则集 $P(X)$, 预期行为规则集规定了合约行为符合预期时的规则, 其中 X 是行为 α 所需满足的安全条件.

定义 1. 合约行为定义为主体使用函数对客体进行操作, 如公式 (1).

$$A = \{action = (s)\ applies\ (f)\ \text{to}\ (obj)\ in\ (env) | s \in subjects,\ f : functions,\ obj : objects,\ env : environment\} \tag{1}$$

其中, *subjects* 为主体集, *functions* 为函数集, *objects* 为客体集, *environment* 为环境状态. 其中主体主要指合约, 函数集主要指合约内置的函数操作 (初始化, 调用, 删除等), 客体集主要指区块链账本或外部库等, 环境主要包括沙箱等, 合约行为定义为合约通过内置的函数对区块链账本进行读或写操作. 合约行为的表现形式定义为系统调用序列. 因此, 对于每个行为单元来说, 主体是系统调用, 客体是文件, 操作是系统调用对文件进行读写等, 环境指内存环境等.

定义 2. 合约行为序列定义为合约在某一行为发生过程中所触发的系统调用序列, 如公式 (2).

$$\alpha = a_1, a_2, ..., a_n \quad a \in system\ call \tag{2}$$

其中, α 代表智能合约的一个行为, $a_1, a_2, ..., a_n$ 表示这一行为执行过程中所触发的系统调用组成的序列. 合约的功能是通过有序的代码段来实现的, 若代码执行的顺序不正确, 该功能就会出错. 因此合约的某一特定功能, 其行为序列是固定的. 系统调用在检测行为是否符合预期方面具有较好的特性, 只需要通过系统调用序列与预期的差异来判定一个合约是否可信. 系统调用是智能合约向区块链系统发出的访问资源的请求, 例如发送交易、读写数据、获取时间戳等, 是合约行为的最小粒度, 它可以提供最细致和完整的行为信息, 因此可以通过比较系统调用序列与预期的差异来判定合约行为的可信. 智能合约代码一般体量较小, 适合通过系统调用描述合约行为过程, 因此, 本文以系统调用作为合约的最小行为单元, 以系统调用序列构建合约行为.

定义 3. 行为可信的定义为行为符合预期, 如公式 (3).

$$\forall_i \in (1 \sim n), \forall_a \in \alpha. \quad Actual(s_0, \alpha, a_i) = Expected(s_0, \alpha, (P(a_i), a_i)) \tag{3}$$

其中, $Actual(s_0, \alpha, a_i)$ 表示行为 α 的实际执行结果, $Expected(s_0, \alpha, (P(a_i), a_i))$ 表示在满足预期行为规则的条件下, 行为 α 的预期执行结果. 如果等式成立, 则表示对于行为 α 中的任意系统调用均满足预期行为规则, 即合约行为符合预期, 因而是可信的. 合约的行为是否可信是由作为其行为单元的系统调用针对预期行为规则 $P(X)$ 的满足性决定的. 合约行为可信不仅讨论每一个行为单元的条件满足性, 还要考虑其执行顺序, 以及合约自身完整性、执行环境等是否满足条件.

3.2 整体架构

智能合约行为可信验证机制的整体架构, 如图 2 所示. 主要包括预处理和主动监控机制两部分, 其中预处理包括控制流图生成、行为特征提取两个模块, 主动监控机制包括合约行为控制、合约行为度量、合约行为判定这 3 个模块.

本文的可信验证机制建立在合约的初始状态可信的安全假设之上.

(1) 控制流图生成模块构建合约的控制流图, 并将其发送给行为特征提取模块.

(2) 行为特征提取模块以控制流图为输入, 输出合约的预期行为规则及基准值并发送到主动监控机制.

(3) 合约行为控制模块拦截合约运行时的系统调用实现对合约的运行时监控, 并根据判定结果实施控制.

(4) 合约行为度量模块依据度量策略对控制机制返回的行为信息度量, 并将度量结果发送到判定模块.

(5) 合约行为判定模块根据可信基准库和度量结果对合约行为进行综合判定.

可信策略管理中心: 对区块链策略制定、下发、维护存储等集中管理的平台, 区块链策略包括但不仅限于节点访问控制、身份验证等.

行为基准库: 主要提供可信基准值数据, 包括但不仅限于智能合约源码、合约控制流图、预期行为规则等基准值数据. 本文行为基准库的获取在确保可信的共识节点上进行合约运行过程监测, 并对合约运行过程加载的静态数据进行完整性度量, 度量完成后将度量值存储到行为基准库中.

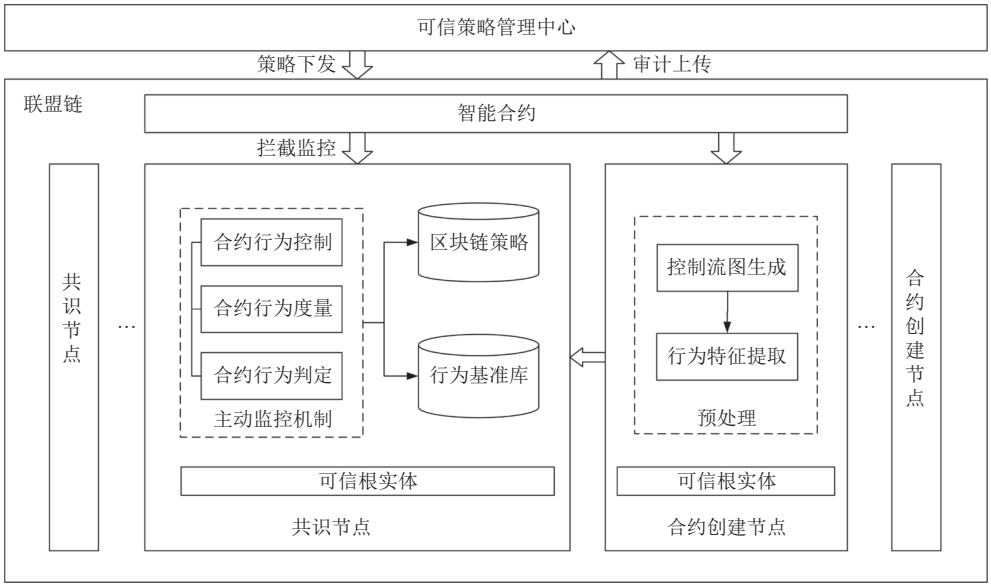


图 2 智能合约行为可信验证机制整体架构

3.3 方案实现

3.3.1 控制流图生成

控制流图生成模块是为了将合约源代码抽象为一种更易于分析和验证的形式, 以便后续模块可以更准确地获取合约的预期行为序列及其规则. 控制流图生成模块的输入是合约源代码, 它描述了合约的逻辑和属性, 输出是合约的控制流图, 它描述了合约的执行过程和状态转换.

控制流图 CFG (control flow graph) 使用图形符号来表示源代码, 反映了源代码的结构和功能性质. 控制流图是一个有向图, 包含许多源代码的执行路径. 控制流图包含作为节点的所有基本块, 以及一些表示跳跃的边, 其中基本块是源代码中的会触发系统调用的语句, 跳跃的边表示语句之间的关系. 然而, 有些边在静态分析阶段无法确定, 需要在动态执行过程中构建[39]. 静态分析利用程序源代码或字节码对代码进行分析而不执行代码, 它通过对代码的语法和结构进行分析, 以推断可能的程序行为. 然而, 静态分析无法获取程序的完整运行时信息, 程序在运行时可能会受到用户输入、外部环境等因素的影响, 这些因素无法在静态分析中完全考虑到. 因此, 静态分析无法确定所有的边, 动态执行过程中边的构建至关重要.

控制流图生成模块首先通过静态分析划分合约功能, 确定各功能对应代码块. 以系统调用为节点, 并将节点之间的关系抽象为边. 根据合约内部调用关系和控制结构生成控制流图, 监控合约运行时的系统关键调用序列以及参数信息, 对合约功能模块进行多次触发, 对静态分析得到的控制流图进行优化, 最后将控制流图发送给行为特征提取模块.

构造合约控制流图主要分为以下步骤.

(1) 扫描合约的源程序, 按照合约的功能需求划分合约功能, 将合约分为初始化函数、调用函数等;

(2) 以合约功能中能够触发系统调用的语句为边界, 划分基本块;

(3) 根据基本块转移关系构造控制流图, 获取合约运行时的系统调用, 补全不确定的控制转移关系, 构建完整的控制流图;

(4) 替换基本块为系统调用, 构建系统调用控制流图.

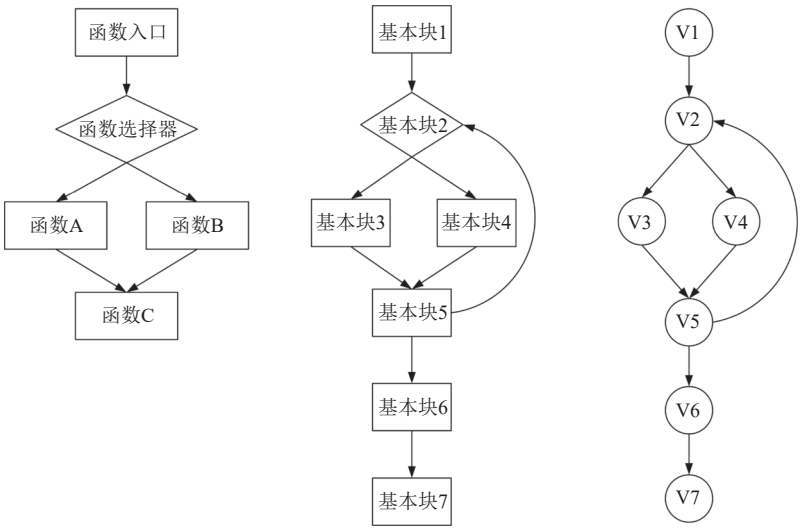
合约的控制流图可以形式化地描述如下.

$CFG = <V, E>$

V =(v_i |每个节点 v 与一个系统调用一一对应)

E =(<v_i, v_j >|表示系统调用 i 到系统调用 j 之间存在控制权转移)

合约的控制流图构造过程如图 3 所示.



(a) 基本功能划分 (b) 功能基本块划分 (c) 系统调用转换

图 3 智能合约控制流图构造过程

3.3.2 行为特征提取

行为特征提取模块的输入是控制流图, 它描述了合约的执行过程和状态转换. 行为特征提取模块的输出是预期行为规则集, 它描述了合约的预期行为和安全策略.

预期行为规则可以用四元组表示 Rules=(F, S, KP, SP), 分别表示以下内容.

(1) 合约的功能模块 (function, F), 即合约可以实现的不同功能.

(2) 合约的预期行为序列 (sequence, S), 即合约在执行每个功能模块时所触发的系统调用及其序列.

(3) 合约的关键参数集 (key parameter, KP), 即合约在执行时触发的每个系统调用的关键参数.

(4) 合约的安全策略 (security policy, SP), 即合约在执行每个系统调用时所需满足的安全条件或约束.

行为特征提取模块的过程如下.

(1) 根据控制流图中的分支结构, 划分合约的不同功能模块, 例如一个众筹合约可能有创建合约、设置目标、设置截止日期、贡献资金、提现资金、退款资金等功能模块.

(2) 对于每个功能模块, 遍历其控制流图中的所有节点, 将按照时间顺序执行的系统调用串联起来作为该功能模块的预期行为序列, 例如创建合约功能模块的预期行为序列可以是: open, write, close 等.

(3) 利用 strace 命令收集合约进程运行过程中正常的系统调用行为, 记录分析各系统调用的参数值范围, 提取关键系统调用参数. 当采用系统调用来表现软件行为时, 需要对各个系统调用进行识别. Linux 系统内存在 393 个系统调用[40], 主要有文件类、进程类、网络类、内核类等. 合约主要对读写等文件类系统调用影响较大, 为了提高效率, 本文只关注文件操作类.

(4) 对于每个合约功能, 分析其可能的安全策略, 例如系统调用访问控制, 节点身份验证, 背书策略验证等.

(5) 各功能模块的预期行为序列、关键参数以及合约安全策略构成合约的预期行为规则集, 并存入可信行为基准库, 一个合约功能绑定一个预期行为规则.

行为特征提取模块从控制流图中提取合约的预期行为规则, 并将其基准值发送到行为基准库以便后续模块可

以判定合约的实时行为是否符合预期. 通过预设的合约预期行为规则, 可以度量合约的任何实时行为, 只要出现合约行为偏离了预期, 都可以被本文方法检测到.

3.3.3 合约行为控制

控制机制是主动监控机制发挥作用的入口, 控制过程拦截智能合约系统调用行为, 获取行为相关的主体、客体、操作、环境等信息, 依据控制策略将信息发送给度量机制进行度量, 并接受判定机制的判定结果, 进行相关的控制. 本文通过 Hook 监控技术在内核层通过 Linux 安全模块 LSM (Linux security module)[41]拦截对系统调用接口的调用, 通过重写钩子函数的方式来对合约运行过程的重点系统调用做主动控制, 将监控点设置到对每一个关键系统调用中. 该方法可以实时监控所有的系统调用, 保证对于合约实际行为监控的完备性.

首先, 共识节点从合约创建交易中获取智能合约源代码, 从合约调用交易中获取合约输入, 在合约执行过程中获取系统调用相关的参数、时间、次数等信息. 其次, 将收集好的相关信息发送给合约行为度量模块, 度量模块将度量结果发送给合约行为判定模块, 控制机制接收合约行为判定模块返回的行为可信判定结果, 并依据判定结果和区块链系统策略实施控制措施.

3.3.4 合约行为控制

度量机制依据度量策略对控制机制传递的合约执行过程中拦截的系统调用的主体、客体、操作、环境等信息按照度量策略进行度量, 并将度量结果发送至判定机制. 本文的度量方法主要采用国密算法 SM3, 度量对象主要包括智能合约行为过程中触发的系统调用以及相关的主体、客体等.

3.3.5 合约行为判定

合约行为判定模块获取合约行为的度量结果, 结合预期行为规则和区块链系统安全策略判定合约行为是否可信, 行为可信的前提有: 节点具有创建、发布、调用、删除合约的权限, 合约及其执行环境可信.

共识节点收集到合约交易, 从预期行为规则中获取合约策略, 判定节点是否有对合约操作的权限. 若没有则抛弃交易, 在执行合约之前对合约运行环境以及合约基准值进行可信验证, 在确保合约和执行环境均可信的前提下, 对合约行为进行可信验证: 共识节点对合约行为监控阶段实时返回的行为信息进行综合判定, 验证合约行为是否符合预期行为规则以及区块链系统安全策略. 若合约行为不符合预期行为规则或不满足区块链系统安全策略, 则抛弃交易并报警, 如果一致则合约行为可信.

合约行为判定模块具体步骤如下.

(1) 获取合约行为 α 的本次系统调用 a, 遍历预期行为规则, 计算行为单元 a 在预期行为规则控制下的预期执行动作 $b=Expected(a,P(a_i))$, 最终有 $b=a$(允许执行 a) 或者 $b=\Lambda$(不允许执行 a). 如果 $b=a$, 则当前系统调用符合预期行为规则, 执行步骤 (2); 若 $b=\Lambda$, 则当前系统调用不可信, 依据控制策略实施控制, 并在检测到其可信性受到破坏后进行报警.

(2) 确定行为 α 下一个需要实时监控的系统调用, 重复步骤 (1). 若交易执行结束, 则合约行为可信, 若当前系统调用不可信, 在检测到其可信性受到破坏后进行报警.

设行为 α 的实际行为序列为 $\alpha=a_1\circ a_2\circ\ldots\circ a_n$ 则根据预期行为规则计算 α 对应预期行为 $\beta=Expected(\alpha,(P(a_i),a_i))$. 由定义 3 可得: $\beta=a_1\circ a_2\circ\ldots\circ a_n$, 其中对于任意的 a_1 有 $P(a_i)$ 成立. 行为 α 是否可信的自动化验证过程如图 4 所示, 图 4 中的圆圈表示区块链状态, 圆圈之间的箭头表示触发的系统调用, 圆圈之间的连线表示实际与预期的状态同步验证关系, 当且仅当实际执行的系统调用与预期相同且满足安全策略时, 允许触发下一个系统调用.

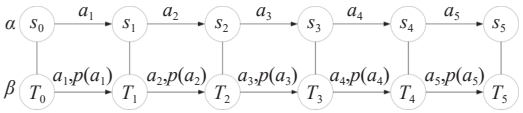


图 4 智能合约行为可信验证

4 安全分析

本文从访问控制、代码漏洞、资源完整性这 3 个方面给出了几个威胁场景下的安全分析, 说明了本文提出的验证机制能够有效地评估合约行为是否符合预期, 保障区块链系统的安全稳定运行.

场景 1: 节点 D 发布或调用一个试图绕过访问控制策略访问未经授权的区块链资源的智能合约.

共识节点收到交易后执行合约, 首先会判断节点 D 是否有发布或调用合约的权限, 然后执行合约. 调用智能合约行为可信验证机制判断合约行为是否可信. 本文研究方案的合约监控模块将合约运行过程中的系统调用以及关键参数实时发送给合约判定机制, 能够及时发现系统调用的关键参数不满足安全策略, 因此该合约行为不可信, 无法被发布或调用.

场景 2: 攻击者试图通过合约逻辑漏洞实现可重入攻击, 攻击者部署了一个恶意合约 Mallory, 并向一个正常的合约 Alice 发送一笔资金. 当 Alice 收到资金后, 会调用 Mallory 的回退函数进行确认. 在回退函数中, Mallory 再次向 Alice 发送资金, 导致 Alice 再次调用 Mallory 的回退函数, 使得 Mallory 可以多次从 Alice 中提取资金.

本文研究方案的预期行为规则包括与被攻击合约绑定的行为序列以及回退函数的调用次数, 决定了重复执行的函数操作是不可信的序列. 当被攻击合约的回退函数调用次数不符合预期行为规则时, 说明被攻击合约的行为不可信, 该方法能够快速有效检测出可重入攻击, 有效防止攻击者通过可重入对区块链进行攻击.

场景 3: 在区块链上部署的合约是经过验证的可信合约, 但是在从区块链拿到本地执行环境的过程中, 合约可能被恶意篡改, 或者执行环境可能被植入恶意漏洞, 导致合约的执行结果发生错误, 合约执行过程中需要从外部获取资源, 攻击者可能恶意篡改外部资源.

本文提出的智能合约行为可信验证机制通过对合约源代码、合约执行环境、外部资源的完整性进行可信度量, 能够及时发现完整性是否被篡改, 一旦检测到合约、执行环境、外部资源完整性遭到破坏, 不允许该合约进行后续操作, 能够保证合约可信执行.

5 实验分析

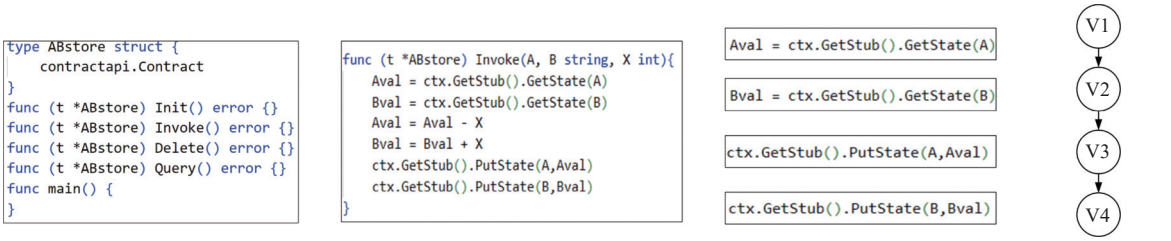
本文基于 Hyperledger Fabric 搭建原型系统, 部署行为可信验证机制. Hyperledger Fabric 是一种具有高度模块化、支持多种编程语言的智能合约、可插拔共识机制的特点的区块链框架, 基于 Hyperledger Fabric 模式构建的商业应用层出不穷, 由国家信息中心牵头发起的国家级区块链平台“区块链服务网络”在国内部署联盟链, 将 Hyperledger Fabric 作为首批适配的区块链项目, 可见 Hyperledger Fabric 在国内也得到广泛认可. 因此, 本文对智能合约的测试针对 Hyperledger Fabric 平台展开.

本文实验环境具体为 64 位 Ubuntu 18.04 操作系统, 内核版本为 4.15.0. 8 GB 内存, Intel(R) Core(TM) i5-8250U CPU @1.60GHz 1.80 GHz, Hyperledger Fabric 版本为 2.3.3, Caliper 版本为 0.4.0, Doker 版本为 20.10.7, Go 版本为 1.16.7, strace 版本为 5.3.

本文针对联盟链 Hyperledger Fabric 中恶意合约对世界状态 4 种类型的事务, 包括只写、只读、读写和删除操作执行不可信攻击的情况. 由于交易只在最后一步才将状态写入世界状态, 因此查询操作不影响区块链账本状态. 本文中只考虑更新操作, 智能合约由多种语言编写, 其中 Go 语言调用 Shim 接口实现核心业务逻辑, Shim 包提供了 stub.PutState 与 stub.GetState 等关键 API 函数来写入和查询区块链键值状态, 这些 API 在执行时会触发 read, write 等文件系统调用. 转账合约在执行时 read 系统调用会打开合约文件、节点身份认证相关文件、账本相关文件等, write 系统调用将更新成功的结果写入区块链账本, 通过 strace 命令追踪合约运行时的系统调用并结合区块链日志分析系统调用参数.

为了评估该机制的有效性, 我们对合约转账行为进行测试, 在 mychaincode 项目中引入了恶意逻辑, 这会导致不符合预期的行为序列, mychaincode 合约源码结构、转账功能源码、转账合约控制流图生成如图 5 所示.

本实验分成两组: 第 1 组功能转移及参数均符合预期, 第 2 组功能转移正确**,** 但参数值不符合行为规则, 分别命名为 contractl、contract2. 实验的结果见表 1.



(a) 基本功能划分 (b) invoke 源码 (c) 基本块划分 (d) 基本块控制流图

图 5 智能合约源码结构与控制流图

表 1 智能合约预期行为与执行监控结果

正常调用	Read	Read	Read	Write
正常调用参数	/fabric-samples/test-network/mychaincode/fan10	/users/Admin@org.example.com/msp /users/Admin@org.example.com/tls	/hyperledger/production/ledgersData/pvtdataStore /hyperledger/production/ledgersData/historyLeveldb /hyperledger/production/ledgersData/stateLeveldb	/hyperledger/production/ledgersData/chains/chains/mychannel/blockfile
contract1	/fabric-samples/test-network/mychaincode/fan10	/users/Admin@org.example.com/msp /users/Admin@org.example.com/tls	/hyperledger/production/ledgersData/pvtdataStore /hyperledger/production/ledgersData/historyLeveldb /hyperledger/production/ledgersData/stateLeveldb	/hyperledger/production/ledgersData/chains/chains/mychannel/blockfile
contract2	/fabric-samples/test-network/mychaincode/fan10	/users/Admin@org.example.com/msp /users/Admin@org.example.com/tls	/hyperledger/production/ledgersData/pvtdataStore /hyperledger/production/ledgersData/historyLeveldb /hyperledger/production/ledgersData/stateLeveldb	/hyperledger/production/ledgersData/chains/index/LOG

表 1 为合约的预期行为系统调用及其参数, 作为基准判定 contractl、contract2 行为是否可信. 实验结果表明, contract1 的实际行为与预期行为相符, 所以 contract1 的行为可信. contract2 的中 write 系统调用的参数与预期不一致, 即 contract2 合约的转账行为不可信. 因此, 本文提出的智能合约行为可信验证机制能够精确判定合约行为的可信性, 并且具有较高的准确性, 具有实用价值.

5.1 对比分析

本文方案与目前已有的研究分别从生命周期、行为、细粒度、可信验证、权限验证、安全性 6 种角度进行对比. 结合表 2 可以看出, 本文所述方案具有以下几个方面的优势.

(1) 生命周期: 本文方案可以对合约部署阶段和合约调用阶段进行验证, 防止经过可信验证的合约上链后在拿到本地环境运行的过程中发生恶意篡改导致合约执行结果错误.

(2) 行为: 从合约行为角度出发, 提出以系统调用为行为单元, 以系统调用序列准确描述合约行为.

(3) 细粒度性: 本文方案对合约运行过程中的系统调用行为进行监测, 能够较为细粒度地对合约部署和运行过程进行可信管控.

(4) 可信验证: 对合约数据完整性以及行为完整性进行度量并通过实时动态监控判定合约是否符合预期.

(5) 权限验证: 区块链根据自身需求制定并维护访问控制策略, 决定合约是否具备创建、调用、删除合约的权限, 系统调用是否具有访问资源的权限. 智能合约自主决定背书策略等是否授予节点调用权限.

(6) 安全性: 本方案的监控机制能够从系统调用层次对合约运行过程的相关行为进行管控, 同时对合约运行过程加载的静态文件进行数据完整性和行为完整性校验, 能够有效防止针对合约的恶意攻击.

表 2 实验数据集

方案	生命周期	行为	细粒度	可信验证	权限验证	安全性
Kalra[18]	×	√	具备	不具备	具备	符合执行
Hu[21]	×	√	不具备	不具备	不具备	异常检测
Chen[23]	×	×	不具备	具备	具备	硬件保护
本文方案	√	√	具备	具备	具备	动态度量

5.2 性能分析

另一方面, 针对安全分析中的 3 种场景我们研究了智能合约行为可信验证机制对 Hyperledger Fabric 区块链系统性能的影响. 我们通过对 mychaincode 合约的转账交易进行测试, 并测试该机制下区块链系统确认交易的延迟. 针对访问控制场景, 在合约执行前匹配预期行为规则, 在合约执行时匹配区块链系统安全策略. 针对合约篡改等资源完整性场景, 在合约执行前匹配基准值与度量值. 针对合约漏洞场景, 在简单、中等、复杂这 3 种合约下进行实验评估该机制的性能.

本文使用能够衡量区块链平台性能的基准工具 Hyperledger Caliper 在单机环境下进行测试, 测试所用数据库为 LevelDB, 测量交易发送请求和接收响应的时间差.

本节采用时间延迟作为本文研究方案的性能评价指标, 其中引起时间延迟的因素包括 4 个部分: (1) 区块链正常交易流程所用的时间 t_1; (2) 捕获关键系统调用所用的时间 t_2; (3) 匹配预期行为规则所用时间 t_3; (4) 系统调用加载客体数据的可信度量时间 t_4. 因此, 总的性能开销为以上 4 部分的和.

智能合约行为可信验证机制相较于普通合约交易过程而言, 其差别在于系统调用执行的监控及可信度量和判定行为. 利用 strace 命令, 结合合约的部署或调用流程对相关系统调用进行统计, 本文选取 read、write 这两类系统调用作为预期行为规则的主要来源. 当上述两类系统调用被捕获, 根据其系统调用参数对其加载的文件进行可信度量, 并根据预期行为规则判定该系统调用行为是否可信. 针对简单、中等、复杂合约, 分别对应系统调用数量为 6、37、100 下进行实验评估. 本文以中等合约为例详细分析合约运行时的性能开销.

(1) 通过性能测试工具 Hyperledger caliper 测试合约正常运行时交易确认的时延 t_1.

(2) 通过 strace 命令获取捕获系统调用的时间 t_2 如表 3 所示.

(3) 为了提高性能, 只对与合约运行相关度较大的文件进行度量, 即合约源文件、节点身份证书、区块链账本等. 本实验调用国密算法 SM3 进行完整性度量, 表 4 给出了客体文件完整性度量的时间消耗 t_3.

(4) 匹配时间 t_4 主要是字符串比较等指针操作, 如表 5 的数据是在实验过程中多次实验得到的平均值.

表 3 系统调用统计

系统调用	次数	耗时 (s)	总耗时 (s)
read	26	0.0000126	0.000327
write	11	0.0000200	0.000220

表 4 完整性度量耗时

文件类型	文件大小 (KB)	耗时 (s)	总耗时 (s)
.go(合约)	3.5	0.000034	0.000034
.pem等(身份认证)	0.78	0.000020	0.000260
blockfile(账本)	105	0.000570	0.000570

表 5 匹配耗时

文件类型	文件大小 (KB)	耗时 (s)	平均耗时 (s)
.go (合约)	3.5	0.00001649	
.pem等 (身份认证)	0.78	0.00001665	0.000016
blockfile (账本)	105	0.00001678	

为了减少误差因素影响, 针对简单、中等、复杂合约各重复进行了 5 次实验, 结果如图 6 所示. 可以计算和分析得出智能合约行为可信验证机制在不同合约情况下的时间消耗情况. 在简单合约下执行消耗的时间相比区块链正常交易增加约 0.0025 s, 在中等合约下执行消耗的时间相比区块链正常交易增加约 0.014 s, 在复杂合约下执行

消耗的时间相比区块链正常交易增加约为 0.038 s. 因此, 该部分实验结果说明了采用智能合约行为可信验证机制之后, 对于区块链性能开销的影响在可接受范围内, 并无显著性能下降.

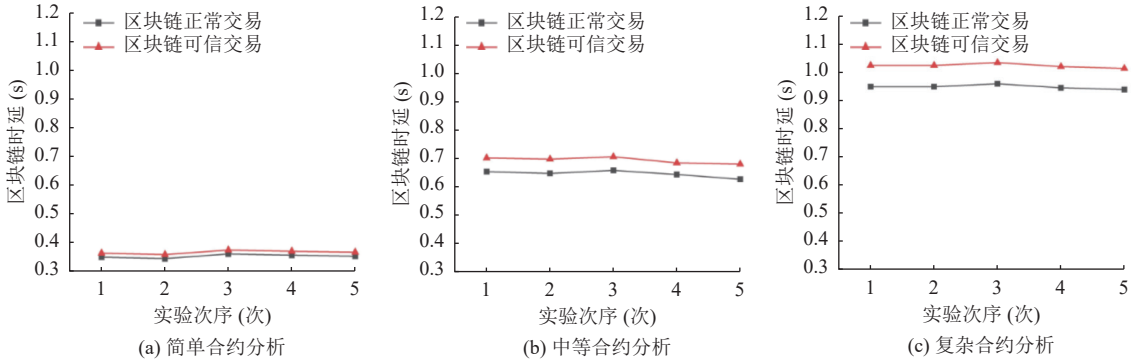


图 6 智能合约调用时延分析

综上所述, 通过对所采用的智能合约行为可信验证机制进行评估, 采用行为可信验证机制后的区块链可信交易时间有所增加, 但增加的时间几乎不会对性能产生影响. 随着合约复杂度的增加, 该机制仍可在确保智能合约行为可信的同时, 保持良好的性能情况.

6 总 结

本文提出一种面向联盟链的智能合约行为可信验证机制, 该方法融合区块链和可信计算技术, 借助静态分析和动态分析提出基于系统调用序列的行为动态度量模型. 同时, 利用联盟链共识的一致性特点在合约完整性未被篡改的情况下, 实现对智能合约行为完整性的主动监控机制. 通过理论分析和实验评估了本文提出的机制, 在 Hyperledger Fabric 平台的实验结果表明, 该机制能够有效地检测出不符合预期的行为序列, 实现了对智能合约行为的可信性判定, 智能合约行为可信验证机制具有良好的效果, 并不会带来过大的性能开销. 与现有工作相比, 能够有效防止恶意合约部署对区块链系统进行恶意攻击, 进一步增强了联盟链系统的安全性. 本文当前侧重对智能合约行为可信性进行研究, 尚未讨论节点的身份可信性问题, 未来将进一步探索研究该问题.

References:

[1] Li XQ, Jiang P, Chen T, Luo XP, Wen QY. A survey on the security of blockchain systems. Future Generation Computer Systems, 2020, 107: 841–853. [doi: 10.1016/j.future.2017.08.020]

[2] Chu HT, Zhang PC, Dong H, Xiao Y, Ji SH, Li WR. A survey on smart contract vulnerabilities: Data sources, detection and repair. Information and Software Technology, 2023: 159: 107221. [doi: 10.1016/j.infsof.2023.107221]

[3] Islam MM, Islam MK, Shahjalal M, Chowdhury MZ, Jang YM. A low-cost cross-border payment system based on auditable cryptocurrency with consortium blockchain: Joint digital currency. IEEE Trans. on Services Computing, 2023, 16(3): 1616–1629. [doi: 10.1109/TSC.2022.3207224]

[4] Dodmane R, Raghunandan KR, Krishnaraj Rao NS, Kallapu B, Shetty S, Aslam M, Jilani SF. Blockchain-based automated market makers for a decentralized stock exchange. Information, 2023, 14(5): 280. [doi: 10.3390/info14050280]

[5] Yang JW, Paudel A, Gooi HB. Compensation for power loss by a proof-of-stake consortium blockchain microgrid. IEEE Trans. on Industrial Informatics, 2021, 17(5): 3253–3262. [doi: 10.1109/TII.2020.3007657]

[6] Xu BY, Xu LD, Wang YX, Cai HM. A distributed dynamic authorisation method for Internet+ medical & healthcare data access based on consortium blockchain. Enterprise Information Systems, 2022, 16(12): 1857–1875. [doi: 10.1080/17517575.2021.1922757]

[7] Ma R, Yang XT, Gao F. Discussion on smart contract under blockchains technology. In: Proc. of the 2022 Int'l Conf. on Industrial IoT, Big Data and Supply Chain (IIoTBDSC). Beijing: IEEE, 2022. 338–342. [doi: 10.1109/IIoTBDSC57192.2022.00069]

[8] Owe O, Fazeldehkordi E. A lightweight approach to smart contracts supporting safety, security, and privacy. Journal of Logical and Algebraic Methods in Programming, 2022, 127: 100772. [doi: 10.1016/j.jlamp.2022.100772]

[9] Li Y, Liu H, Yang ZQ, Wang B, Ren Q, Wang L, Chen BD. Protect your smart contract against unfair payment. In: Proc. of the 2020 Int'l Symp. on Reliable Distributed Systems (SRDS). Shanghai: IEEE, 2020. 61–70. [doi: 10.1109/SRDS51746.2020.00014]

[10] Xiang DM, Lin YC, Nie LM, Zheng YW, Xu ZZ, Ding ZH, Liu Y. An empirical study of attack-related events in DeFi projects development. Empirical Software Engineering, 2024, 29(2): 49. [doi: 10.1007/s10664-024-10447-7]

[11] Shen CX. To create a positive cyberspace by safeguarding network security with active immune trusted computing 3.0. Journal of Information Security Research, 2018, 4(4): 282–302 (in Chinese with English abstract). [doi: 10.3969/j.issn.2096-1057.2018.04.001]

[12] Chen HS, Pendleton M, Njilla L, Xu SH. A survey on ethereum systems security: Vulnerabilities, attacks, and defenses. ACM Computing Surveys (CSUR), 2020, 53(3): 67. [doi: 10.1145/3391195]

[13] Cui ZQ, Yang HW, Chen X, Wang LZ. Research progress of security vulnerability detection of smart contracts. Ruan Jian Xue Bao/Journal of Software, 2024, 35(5): 2235–2267 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/7046.htm [doi: 10.13328/j.cnki.jos.007046]

[14] Wang W, Song JJ, Xu GQ, Li YD, Wang H, Su CH. ContractWard: Automated vulnerability detection models for ethereum smart contracts. IEEE Trans. on Network Science and Engineering, 2020, 8(2): 1133–1144.

[15] State Administration for Market Regulation, Standardization Administration. GB/T 22239-2019 Information security technology-baseline for classified protection of cybersecurity. Beijing: Standards Press of China, 2019. 92 (in Chinese).

[16] Zheng PL, Zheng ZB, Luo XP. Park: Accelerating smart contract vulnerability detection via parallel-fork symbolic execution. In: Proc. of the 31st ACM SIGSOFT Int'l Symp. on Software Testing and Analysis. Virtual: ACM, 2022. 740–751. [doi: 10.1145/3533767.3534395]

[17] Hu K, Zhu J, Ding Y, Bai XM, Huang JH. Smart contract engineering. Electronics, 2020, 9(12): 2042. [doi: 10.3390/electronics9122042]

[18] Kalra S, Goel S, Dhawan M, Sharma S. ZEUS: Analyzing safety of smart contracts. In: Network and Distributed Systems Security (NDSS) Symp. 2018. San Diego, 2018. 1–12. [doi: 10.14722/ndss.2018.23082]

[19] Lv PH, Wang Y, Wang YZ, Zhou QH. Potential risk detection system of hyperledger fabric smart contract based on static analysis. In: Proc. of the 2021 IEEE Symp. on Computers and Communications (ISCC). Athens: IEEE, 2021. 1–7. [doi: 10.1109/ISCC53001.2021.9631249]

[20] Ding MJ, Li PR, Li SS, Zhang H. HFContractFuzzer: Fuzzing hyperledger fabric smart contracts for vulnerability detection. In: Proc. of the 25th Int'l Conf. on Evaluation and Assessment in Software Engineering. Trondheim: ACM, 2021. 321–328. [doi: 10.1145/3463274.3463351]

[21] Hu T, Liu XL, Chen T, Zhang XS, Huang XM, Niu WN, Lu JZ, Zhou K, Liu Y. Transaction-based classification and detection approach for Ethereum smart contract. Information Processing & Management, 2021, 58(2): 102462. [doi: 10.1016/j.ipm.2020.102462]

[22] Liu ZG, Qian P, Wang XY, Zhuang Y, Qiu L, Wang X. Combining graph neural networks with expert knowledge for smart contract vulnerability detection. IEEE Trans. on Knowledge & Data Engineering, 2023, 35(2): 1296–1310. [doi: 10.1109/TKDE.2021.3095196]

[23] Chen P, Shi PC, Xu J, Fu X, Li LH, Zhong T, Xiang LL, Kong JZ. TeeSwap: Private data exchange using smart contract and trusted execution environment. In: Proc. of the 23rd Int'l Conf. on High Performance Computing & Communications; the 7th Int'l Conf. on Data Science & Systems; the 19th Int'l Conf. on Smart City; the 7th Int'l Conf. on Dependability in Sensor, Cloud & Big Data Systems & Application. Haikou: IEEE, 2021. 237–244. [doi: 10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00057]

[24] Cheng R, Zhang F, Kos J, He W, Hynes N, Johnson N, Juels A, Miller A, Song D. Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts. In: Proc. of the 2019 IEEE European Symp. on Security and Privacy (EuroS&P). Stockholm: IEEE, 2019. 185–200. [doi: 10.1109/EuroSP.2019.00023]

[25] Yan Y, Wei CZ, Guo XP, Lu XM, Zheng XF, Liu Q, Zhou CH, Song XY, Zhao BR, Zhang H, Jiang GF. Confidentiality support over financial grade consortium blockchain. In: Proc. of the 2020 ACM SIGMOD Int'l Conf. on Management of Data. Portland: ACM, 2020. 2227–2240. [doi: 10.1145/3318464.3386127]

[26] Brandenburger M, Cachin C, Kapitza R, Sorniotti A. Trusted computing meets blockchain: Rollback attacks and a solution for Hyperledger Fabric. In: Proc. of the 38th Symp. on Reliable Distributed Systems (SRDS). Lyon: IEEE, 2019. 324–333. [doi: 10.1109/SRDS47363.2019.00045]

[27] Keleket Goma CJY, Yi WZ, Wang J. A lightweight trusted execution environment construction method for fabric chaincode based on SGX. Netinfo Security, 2022, 22(7): 73–83 (in Chinese with English abstract). [doi: 10.3969/j.issn.1671-1122.2022.07.009]

[28] Hofmeyr SA, Forrest S, Somayaji A. Intrusion detection using sequences of system calls. Journal of Computer Security, 1998, 6(3): 151–180. [doi: 10.3233/JCS-980109]

[29] Forrest S, Hofmeyr SA, Somayaji A, Longstaff TA. A sense of self for Unix processes. In: Proc. of the 1996 IEEE Symp. on Security and Privacy. Oakland: IEEE, 1996. 120–128. [doi: 10.1109/SECPRI.1996.502675]

[30] Onwuzurike L, Mariconti E, Andriotis P, De Cristofaro E, Ross G, Stringhini G. MaMaDroid: Detecting android malware by building

Markov chains of behavioral models (extended version). ACM Trans. on Privacy and Security (TOPS), 2019, 22(2): 14. [doi: 10.1145/3313391]

[31] Mishra P, Khurana K, Gupta S, Sharma MK. VMAnalyzer: Malware semantic analysis using integrated CNN and bi-directional LSTM for detecting VM-level attacks in cloud. In: Proc. of the 12th Int'l Conf. on Contemporary Computing (IC3). Noida: IEEE, 2019. 1–6. [doi: 10.1109/IC3.2019.8844877]

[32] Cai XQ, Deng Y, Zhang L, Shi JC, Chen Q, Zheng WL, Liu ZQ, Long Y, Wang K, Li C, Guo MY. The principle and core technology of blockchain. Chinese Journal of Computers, 2021, 44(1): 84–131 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2021.00084]

[33] Szabo N. Formalizing and securing relationships on public networks. First Monday, 1997, 2(9). [doi: 10.5210/fm.v2i9.548]

[34] Melo C, Oliveira F, Dantas J, Araujo J, Pereira P, Maciel R, Maciel P. Performance and availability evaluation of the blockchain platform hyperledger fabric. The Journal of Supercomputing, 2022, 78(10): 12505–12527. [doi: 10.1007/s11227-022-04361-2]

[35] Feng DG, Liu JB, Qin Y, Feng W. Trusted computing theory and technology in innovation-driven development. Scicientia Sinica Informationis, 2020, 50(8): 1127–1147 (in Chinese with English abstract). [doi: 10.1360/SSI-2020-0096]

[36] TCG Group. TCG specification architecture overview. TCG Specification Revision, 2007, 1(4): 1–24.

[37] Huang HX, Zhang JB, Yuan YL, Wang X. Research on trusted startup of virtual machine based on non-interference theory. Ruan Jian Xue Bao/Journal of Software, 2023, 34(6): 2959–2978 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6507.htm [doi: 10.13328/j.cnki.jos.006507]

[38] Qu YW. Software Behavior. Beijing: Publishing House of Electronics Industry, 2004 (in Chinese).

[39] Huo X, Li M, Zhou ZH. Control flow graph embedding based on multi-instance decomposition for bug localization. In: Proc. of the 34th AAAI Conf. on Artificial Intelligence. New York: AAAI, 2020. 4223–4230. [doi: 10.1609/aaai.v34i04.5844]

[40] Bagherzadeh M, Kahani N, Bezemer CP, Hassan AE, Dingel J, Cordy JR. Analyzing a decade of Linux system calls. Empirical Software Engineering, 2018, 23(3): 1519–1551. [doi: 10.1007/s10664-017-9551-z]

[41] Isohara T, Takemori K, Miyake Y, Qu N, Perrig A. LSM-based secure system monitoring using kernel protection schemes. In: Proc. of the 2010 Int'l Conf. on Availability, Reliability and Security. Krakow: IEEE, 2010. 591–596. [doi: 10.1109/ARES.2010.48]

附中文参考文献:

[11] 沈昌祥. 用主动免疫可信计算 3.0 筑牢网络安全防线营造清朗的网络空间. 信息安全研究, 2018, 4(4): 282–302. [doi: 10.3969/j.issn.2096-1057.2018.04.001]

[13] 崔展齐, 杨慧文, 陈翔, 王林章. 智能合约安全漏洞检测研究进展. 软件学报, 2024, 35(5): 2235–2267. http://www.jos.org.cn/1000-9825/7046.htm [doi: 10.13328/j.cnki.jos.007046]

[15] 国家市场监督管理总局, 国家标准化管理委员会. GB/T 22239-2019 信息安全技术 网络安全等级保护基本要求. 北京: 中国标准出版社, 2019. 92.

[27] Keleket Goma CJY, 易文哲, 王鹃. 一种基于 SGX 的轻量 Fabric 链码可信执行环境构建方法. 信息网络安全, 2022, 22(7): 73–83. [doi: 10.3969/j.issn.1671-1122.2022.07.009]

[32] 蔡晓晴, 邓尧, 张亮, 史久琛, 陈全, 郑文立, 刘志强, 龙宇, 王堃, 李超, 过敏意. 区块链原理及其核心技术. 计算机学报, 2021, 44(1): 84–131. [doi: 10.11897/SP.J.1016.2021.00084]

[35] 冯登国, 刘敬彬, 秦宇, 冯伟. 创新发展中的可信计算理论与技术. 中国科学: 信息科学, 2020, 50(8): 1127–1147. [doi: 10.1360/SSI-2020-0096]

[37] 黄浩翔, 张建标, 袁艺林, 王晓. 基于无干扰理论的虚拟机可信启动研究. 软件学报, 2023, 34(6): 2959–2978. http://www.jos.org.cn/1000-9825/6507.htm [doi: 10.13328/j.cnki.jos.006507]

[38] 屈延文. 软件行为学. 北京: 电子工业出版社, 2004.

张建标(1969－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为可信计算, 系统安全, 云安全, 区块链技术.

张兆乾(1992－), 男, 博士, 工程师, 主要研究领域为可信计算, 能源区块链, 访问控制.

康双(1999－), 女, 硕士生, 主要研究领域为可信计算, 区块链技术.

刘德田(1997－), 男, 博士生, 主要研究领域为区块链技术, 联邦学习, 信息安全.