doi :10.3969/j.issn.1671-1122.2022.07.009

一种基于 SGX 的轻量 Fabric 链码可信执行环境构建方法

KELEKET GOMA Christy Junior Yannick[1,2]，易文哲 [1,2]，王鹃 [1,2]

（1. 武汉大学国家网络安全学院，武汉 430072；2. 武汉大学空天信息安全与可信计算教育部重点实验室，武汉 430072）

摘　要：Hyperledger Fabric 是一个开源分布式账本平台，其不仅拥有公有链防篡改、分布式记账的特点，还具有身份识别、数据保密、低延迟、高吞吐率等优点。传统 Fabric 架构中的链码缺乏安全执行环境，其容器执行环境会带来隐私泄露风险，而现有的智能合约隐私保护方案无法适用于 Go 语言链码架构，且性能开销较大。因此，文章提出一种基于 SGX 的轻量 Fabric 链码可信执行环境构建方法及 E-Fabric 架构，搭建支持原生 Go 语言的可信镜像和容器，为链码创建可信执行环境，并通过远程认证协议验证链码是否可信。理论评估和数据测试结果表明，SGX Enclave 的构建会适当增加开销，与原 Fabric 架构相比，E-Fabric 的延迟升高了 8% 左右，吞吐率下降了 4% 左右，但整体性能达到原网络的 94%，并且具有较小的可信计算基和更好的安全性。

关键词：区块链；Hyperledger Fabric；链码；可信执行环境；Intel SGX

中图分类号：TP309　**文献标志码：**A　**文章编号：**1671-1122（2022）07-0073-11

中文引用格式：KELEKET GOMA C J Y，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 [J]. 信息网络安全，2022，22（7）：73-83.

英文引用格式：KELEKET GOMA C J Y, YI Wenzhe, WANG Juan. A Lightweight Trusted Execution Environment Construction Method for Fabric Chaincode Based on SGX[J]. Netinfo Security, 2022, 22(7): 73-83.

A Lightweight Trusted Execution Environment Construction Method for Fabric Chaincode Based on SGX

KELEKET GOMA Christy Junior Yannick[1,2], YI Wenzhe[1,2], WANG Juan[1,2]

(1. *School of Cyber Science and Engineering*, *Wuhan University*, *Wuhan* 430072, *China*; 2. *Key Laboratory of Aerospace Information Security and Trusted Computing of Ministry of Education*, *Wuhan University*, *Wuhan* 430072, *China*)

Abstract: Hyperledger Fabric is an open source distributed ledger platform, which not only takes advantage of the tamper-proof and distributed accounting features of the public chain, but also incorporates advantages such as identity recognition, data confidentiality, low latency and high throughput. The chaincode in the traditional Fabric architecture lacks a

收稿日期：2022-03-05

基金项目：国家自然科学基金 [61872430]；国家重点研发计划 [2014CB340600]；湖北省重点研发项目 [2020BAB101]

作者简介：KELEKET GOMA Christy Junior Yannick（1986—），男，刚果共和国，硕士研究生，主要研究方向为区块链安全和可信计算；易文哲（2001—），男，湖北，硕士研究生，主要研究方向为区块链安全、可信计算、系统与网络安全；王鹃（1976—），女，湖北，教授，博士，主要研究方向为系统与软件安全、可信计算、人工智能应用、云计算、物联网安全。

通信作者：王鹃 jwang@whu.edu.cn

secure execution environment, and its container operating environment will bring the risk of privacy leakage. And the existing smart contract privacy protection scheme cannot be applied to the Go language chaincode architecture, and there are defects such as high performance overhead. Therefore, a method and framework were proposed for constructing a lightweight trusted execution environment for Fabric chaincode based on SGX-called E-Fabric, which built trusted images and containers that supported native Go language, created trusted execution environment for chaincode, and verified whether the chaincode was trusted through the remote attestation protocol. Theoretical evaluations and experimental tests show that the creating of the SGX Enclave will increase the overhead. Compared with the original Fabric network, the E-Fabric's latency increases by about 8%, the throughput decreases by about 4%, and the overall performance can reach 94% of the original Fabric network. At the same time, E-Fabric has a small trusted computing base and stronger security.

Key words: blockchain; Hyperledger Fabric; chaincode; trusted execution environment; Intel SGX

0 引言

区块链和分布式账本具有去中心化、不可篡改、可追溯和透明性等特点，是当前研究的热点[1-3]。在区块链上执行特定功能的程序被称为智能合约，它不需要第三方即可准确执行，可以保证交易的正确性。智能合约是区块链的核心技术之一，在应用过程中智能合约代码漏洞和用户隐私泄露会给人们带来巨大的损失。例如，2020年借贷协议dForce由于DeFi智能合约漏洞损失2500万美元。随着区块链用户量和数据量的不断增加，交易场景越来越复杂，这对区块链智能合约的设计和安全防护提出了新挑战。

在常用的开源区块链平台Fabric中，链码（chaincode）由peer节点批准和安装，其功能完全透明和公开。为了方便用户操作，Fabric使用基于Docker容器的隔离运行环境，peer节点和链码运行在独立的Docker容器中，相互隔离。Fabric底层运行环境存在隐私泄露风险，如果恶意用户进入容器，就可以轻易获取链码运行数据和用户隐私，甚至干扰链码运行。例如，在转账时链码会获取用户账户、地址、余额等信息，而这些信息在处理过程中是不受保护的，不受保护的运行环境很容易造成隐私泄露，给用户带来安全隐患。

目前，许多学者使用密码技术增强区块链智能合约的安全性和隐私性，如安全多方计算、同态加密、零知识证明等[4-7]。但这些技术发展不成熟，目前使用这些技术产生的开销较大。可信执行环境是一种高效隐私计算解决方案，为保证区块链智能合约的安全性提供了一种新途径。Intel SGX[8-10]是目前使用较广泛的一种可信执行环境，它可以为应用程序创建一个隔离且受信任的空间，该空间被称为飞地（Enclave）。Enclave可以确保运行的数据和程序不被外界获取和恶意篡改，即使基本输入输出系统（Basic Input Output System，BIOS）、操作系统（Operating System，OS）、Hypervisor和超级管理员也无法访问数据和代码。Intel SGX可以为智能合约提供安全隔离的运行空间，首先，使用Intel SGX的封装技术将Enclave与可信端之间的数据进行加密；然后，将加密后的数据在Enclave中解密和处理；最后，Enclave将计算结果写入区块链，从而保证智能合约执行过程中数据的隐私安全。区块链系统十分复杂，因此针对智能合约构建高效易用的可信执行环境具有重要意义。

目前，已有研究人员提出一些基于可信执行环境的方案，以保证区块链智能合约安全和隐私安全。例如，Coco架构[11]通过共识机制、分布式账本和可信执行环境保护Intel SGX中的智能合约，并将所有节点放入SGX Enclave中运行，使得数据只在可信环境之间传播，从而保证数据传输的安全性。Ekiden是基于以太坊开发的可信智能合约平台，该平台通过在可信执行环境节点中运行智能合约，以确保智能合约的可靠性和可扩展性[12]。FPC（Fabric Private Chaincode）平台[13]基于Hyperledger

Fabric 创建分类账本 Enclave 和链码 Enclave，并添加了 Enclave 注册机构保证账本、链码的正确性[14,15]。

在上述方案中，Ekiden 平台基于以太坊进行开发，需要使用以太币；Coco 只适用于特定的区块链，不能在私有链 Hyperledger Fabric 中使用；FPC 平台虽然基于 Fabric 进行开发，但其为了引入 SGX Enclave，使用 C 语言和 Go 语言重构了整个 Fabric 架构，与现有容器运行环境区别较大，开发代价很大。综上所述，针对 Hyperledger Fabric 链码隐私安全问题，目前缺少轻量、高效的可信执行环境架构。

为了保证链码运行时的隐私安全，本文提出一种新的轻量 Fabric 链码可信执行环境构建方法和 E-Fabric 架构。首先，E-Fabric 为 Fabric 链码创建一个轻量 Enclave，确保数据处理过程中无法从外部访问数据；然后，E-Fabric 仅将链码放入 Enclave 中，使 Enclave 加载的数据量最少，并且可以在不更改 Fabric 原编程语言（Go 语言）的情况下与 peer 节点高效交互。peer 节点通过 SGX 远程认证功能与链码 Enclave 确认身份，保证链码在 Enclave 中安全运行。E-Fabric 的性能可以达到原始 Fabric 网络的 94%，并且具有更好的隐私保护能力。

1 相关技术问题

1.1 Hyperledger Fabric

Hyperledger Fabric 是一个为企业应用程序设计的开源分布式账本平台，其通过成员服务提供者（Membership Service Provider，MSP）进行成员身份管理和验证，并支持模块化共识协议和用户许可政策[14,15]。Fabric 使用 Java、Go 和 JavaScript 等通用编程语言进行编码，Fabric 中的智能合约称为链码，可以执行特定功能的交易。

Hyperledger Fabric 网络结构如图 1 所示，Fabric 网络由客户端（client）、节点（peer）、排序服务（orderer）和账本（ledger）4 个部分组成，具体交易流程如下。

1）交易请求

客户端通过向一个或多个背书节点发送交易请求来调用链码，该请求包括客户端身份、交易负载、参数和链码标识符[14]等信息。

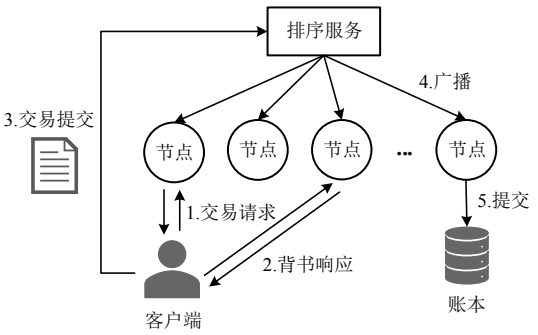

图 1 Hyperledger Fabric 网络结构

2）背书响应

首先，背书节点检查提案的有效性和权威性，只有在检查通过时，节点才会执行链码并模拟请求。其中链码使用客户端传递的参数执行特定操作，不会将结果写入账本或更改世界状态。然后，节点将链码执行过程中的状态变化记录在读写集中，并将背书结果返回客户端。

3）交易提交

客户端收集多个节点的背书结果，直至满足链码设置的背书策略才停止收集，将收集的背书结果组装成一个交易并将此交易提交给排序服务。

4）排序和广播

排序服务不验证也不执行交易，它只收集每个通道上所有已提交的事务，将多个事务批处理到一个区块中，并将区块广播给所有节点。

5）验证和提交

节点接收区块后对其进行验证。节点首先验证交易是否满足背书策略，然后进行读写冲突验证[14]。当两个验证都通过时，节点将新区块写入分类账本并更新世界状态；当没有通过两个验证时，区块不会改变账本[16,17]。

1.2 Intel SGX

Intel SGX[8]可以在用户空间中为应用程序提供一个可信的执行环境，这个可信的执行环境被称为 Enclave，它可以保证数据的机密性、完整性不被恶意程序窃取或篡改。Enclave 是一个隔离的空间，BIOS、OS、Hypervisor

和最高权限的用户都无法访问Enclave中的数据和代码。SGX的可信计算基（Trusted Computing Base，TCB）只包含硬件，避免软件TCB产生的漏洞和威胁，以便更好地保证运行程序的安全性[18]。

SGX提供封装功能，让Enclave中的加密数据能够持久存储在磁盘中[19]。为了防止由于Enclave被破坏（如程序退出或电源故障）而丢失数据的情况发生，所有数据都在Enclave外进行加密存储，Enclave中的软件可以使用封装密钥进行加/解密[20]。

SGX使用认证机制向其他程序证明本地程序在Enclave中安全运行[20]，SGX支持本地认证和远程认证两种认证机制。针对同一台机器上的两个Enclave，首先使用本地认证验证对方是否在同一个TCB平台上运行，然后交换密钥和传输机密数据。针对Enclave和第三方不在同一个TCB平台上的情况，使用远程认证进行身份证明。一般而言，远程认证的目标是让硬件实体、硬件与软件的组合获得远程服务提供商的信任，以便服务提供商为客户端提供机密信息[21]。

1.3 智能合约安全和隐私泄露问题

多数区块链平台使用智能合约实现复杂的交易，如以太坊和Fabric，但这些平台都缺少相应的隐私保护机制。在以太坊中，合约数据和运行代码是公开的，矿工通过模拟执行智能合约，可以得知运行的数据和结果[22]。例如，用户建立一个智能合约，在某个时间将一定量的以太币转移给另一个用户。如果攻击者有其中一个用户的背景信息，就可以将用户信息与真实身份对应起来，导致用户隐私泄露[23-26]。

目前，区块链系统未实现智能合约隐私保护[12]。为了保护隐私，用户一般利用假名标识身份，但这种方式的安全性较弱，并且已有研究通过分析加密货币的事务图结构演示去匿名攻击，打破了匿名化，说明此方式无法较好地保护隐私信息。区块链智能合约具有强大的功能和表现力，但缺少交易隐私保护机制[10]。隐私安全从根本上限制了智能合约的应用范围，因此如何保证智能合约的隐私安全成为一个重要的研究方向。

Fabric使用身份混合器机制[27]实现事务发起者的匿名性和不可链接性，但其不能保证智能合约运行过程和运行环境的机密性。文献[16]分析了Fabric的容器运行结构，并指出当前链码运行环境的脆弱性。为了保证程序之间互不干扰，Fabric将节点、链码、排序服务放入隔离的Docker容器中运行。Fabric创建容器时依靠标签选取镜像，因此攻击者可以创造带后门的镜像，并使用Fabric镜像标签进行假冒。一旦链码运行在后门容器中，攻击者便可以轻松地进入容器内部，从而获取链码运行的隐私数据和结果，甚至破坏链码运行。

在区块链中，智能合约执行过程中的隐私安全十分重要[28]。例如，在使用区块链技术进行非公开选举时，选民的个人信息和投票对象均需要保密，利用智能合约统计选举人的票数，直至最后统计结果才可公开。目前，智能合约没有可信的执行环境，在运算时不受保护，因此容易发生运算数据被窃取和隐私泄露事件。若运算时票数信息被泄露，则敌对方可能采取其他措施，对所有选举人产生影响。此外，在非公开拍卖场景中，报价信息也需要保密。

文献[13]针对基于SGX的智能合约隐私保护方法，提出可信环境构建的性能问题。对于Fabric来说，将所有节点、链码、排序服务甚至账本都放入SGX中是很困难的，因为SGX Enclave内存有限，所以程序过大会导致频繁换页操作，使得整体性能下降。可信环境有一个非常重要的指标——TCB，TCB越小则相应的错误就越少，系统也就越安全。因此，考虑TCB和智能合约性能两方面，系统设计时应尽量减少Enclave内部装载的程序和数据，以保证TCB最小化，使隐私安全性更好。

2 E-Fabric 系统设计与实现

Hyperledger Fabric有多种运行环境，较常用的是基于Docker的隔离容器运行环境，在这种运行环境下，Fabric在独立容器中分别运行节点、排序服务和链码，存在隐私泄露风险。因此，针对基于Docker的隔离容器运行环境，本文提出了一种基于SGX的轻量Fabric

链码可信执行环境构建方法和E-Fabric架构。E-Fabric支持用户直接把基于Go语言编写的智能合约放入SGX Enclave中，不需要人工将智能合约转换成SGX的语言格式，保证了Fabric链码运行环境的安全性。

2.1 系统架构

E-Fabric架构基于原Hyperledger Fabric架构设计，如图2所示，集成SGX驱动程序后在运行时环境中进行扩展，依赖EGo[17]生成链码Enclave。

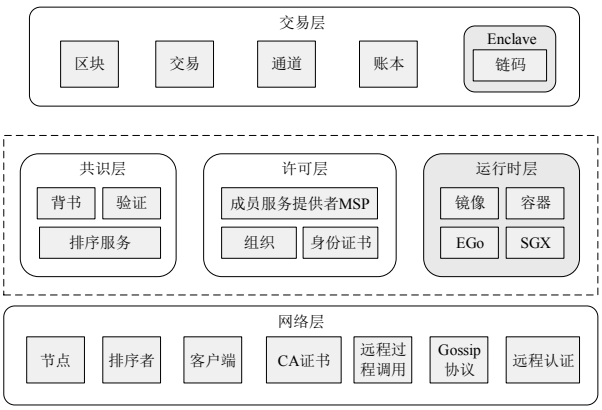

图 2 E–Fabric 架构

在网络层，排序服务、客户端和节点运行在不可信环境中，链码运行在可信环境中。排序服务、客户端、节点和链码之间都使用gRPC进行连接，区别在于排序服务、客户端和节点之间使用跨容器连接，只验证双方身份，不验证环境；而链码需要从Enclave内部向节点发送连接请求，并通过远程认证验证链码运行环境。

共识层和许可层与原始结构一致。用户依靠许可层的证书和MSP进行网络访问和通信，存储区块和更新分类账本操作依赖于共识层的背书、排序和验证。E-Fabric在运行时层进行扩展，使用SGX和EGo为链码提供可信环境，保证链码执行过程中的隐私安全。在交易层，根据运行时层创建的Enclave执行链码。整个网络在底层进行扩展，对外接口没有变化，因此用户感知不到差异。

综上所述，架构中唯一可信部分是在Enclave中运行的链码，其他节点、排序服务和分类账本都被认为是不可信的，这些不可信的部分依靠Fabric MSP相互认证，并且在Enclave外运行。Enclave用于执行用户链码，并与其他节点、链码和排序服务隔离。原始结构在不同Docker容器中分别运行节点和链码，因此需要在容器中使用SGX和EGo创建Enclave。

2.2 基于 SGX 和 EGo 的镜像构建

E-Fabric为每个peer、chaincode、orderer创建单独的Docker容器，从而提供隔离的运行环境。根据容器内运行程序的不同，使用不同的镜像创建Docker容器，E-Fabric核心镜像如表1所示。

表 1 E–Fabric 核心镜像

镜像名称	父镜像	基本功能
fabric-peer	alpine:3.10	peer 节点镜像，安装 peer 相关文件，用于生成 peer 运行容器
fabric-orderer	alpine:3.10	排序节点镜像，安装 orderer 相关文件，用于生成 orderer 运行容器
efabric/base	ubuntu:20.04	E-Fabric 可信基础镜像，作为链码编译、运行的父镜像
ego-baseos	efabric/base:0.1	Go 链码可信运行镜像，用于生成链码运行容器
ego-ccenv	efabric/base:0.1	Go 链码可信编译镜像，在链码实例化过程中作为默认编译环境将链码编译为二进制文件

在表1中，fabric-peer、fabric-orderer镜像基于alpine镜像构建，具有轻量、高效等特点，用于构建peer、orderer运行容器。alpine镜像只支持基础Linux指令，不能满足SGX的复杂安装需求和运行需求，因此针对链码编译和运行镜像，本文将efabric/base:0.1作为父镜像构建。efabric/base:0.1镜像基于ubuntu:20.04镜像构建，其集成了SGX、EGo环境，但本身不具备SGX驱动，容器运行时需要从主机上挂载/dev/isgx或/dev/sgx驱动。链码编译镜像ego-ccenv提供Go语言编译环境，链码运行镜像ego-baseos用于创建链码运行容器。

E-Fabric镜像和容器生成过程如图3所示。首先创建peer、orderer容器，并启动对应程序。链码经过所有peer节点批准安装后打包放入peer容器内，peer容器负责链码包的解压、编译、容器创建。然后peer节点使用ego-ccenv镜像环境编译链码，并传入ego-baseos镜像环境中，在创建链码容器后启动链码Enclave，最后让链码在Enclave中安全运行。

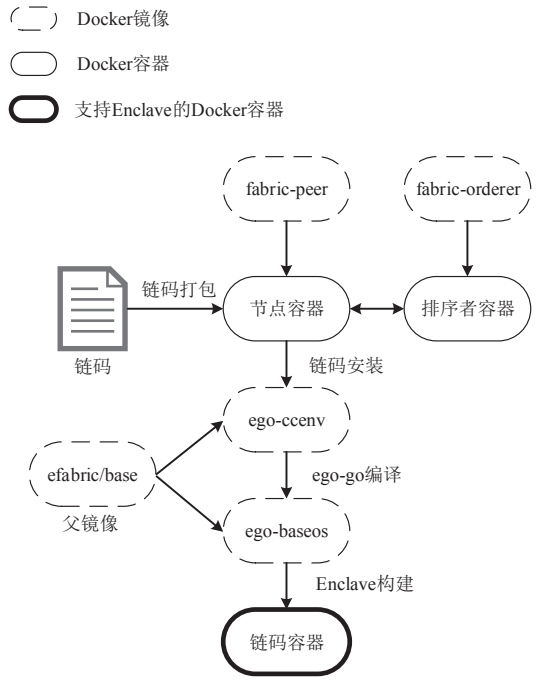

图 3 E-Fabric 镜像和容器生成过程

链码采用Go语言编写，使用EGo进行可信编译和Enclave环境创建。EGo[17]是由Edgeless Systems GmbH开发的架构，负责为Go语言创建可信执行环境，其允许用户在Enclave中直接运行Go语言程序。EGo基于Intel SGX和Open Enclave构建，可以较方便地移植到其他硬件平台中，并且能在具有SGX功能的Intel处理器上轻松创建EGo Enclave。EGo是轻量级的，它只加载Go运行时所需的内容到Enclave中，保持TCB较小[29,30]。

因为Enclave内外隔离的特性，Enclave中的程序无法使用外部挂载和环境变量，所以运行的链码无法获取密钥和证书。E-Fabric使用Enclave配置文件设置挂载、环境变量和嵌入文件，该配置文件类似于SGX中的XML文件。在启动链码容器后，首先创建配置文件并设置Enclave栈大小等信息；然后将环境变量传递给Enclave，包括链码id、客户端证书密钥路径和名称、根证书路径和本地MSP id等；最后设置客户端身份证书、密钥、节点证书等文件挂载，便于Enclave中的链码与外部节点互相认证身份。

2.3 链码安全执行

E-Fabric网络节点和链码间通信过程如图4所示。背书节点安装链码后创建链码容器，链码容器依赖SGX驱动程序和EGo创建链码Enclave，链码在Enclave中安全运行。链码与特定地址的节点建立gRPC连接，以便于容器间的通信。peer节点与链码容器进行远程认证并交换密钥，以确认链码是否在Enclave中安全运行。当产生交易时，背书节点向连接的链码发送交易请求，并对交易请求进行加密。链码接收请求参数后，读取节点容器中的账本和世界状态信息，产生运行结果并将其返回至节点。节点经过背书、分发、验证等操作后，若验证通过，则将区块写入链中并更新世界状态。

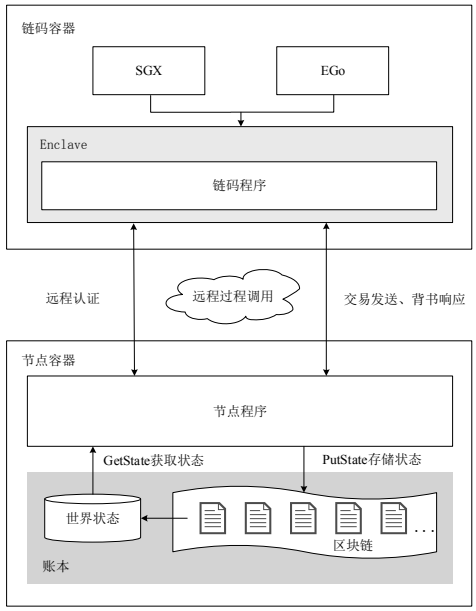

图 4 E-Fabric 网络节点和链码间通信过程

链码使用Go语言进行编写、编译，并在Enclave中运行，不需要划分可信和不可信代码部分，极大地方便了用户使用Go语言编写链码。用户在使用E-Fabric架构时，只需要使用原Fabric的命令行和链码，在使用体验上感知不到差异。

E-Fabric可信接口如表2所示。在远程认证阶段，E-Fabric使用3个应用程序编程接口（Application Programming Interface，API）进行认证。在peer节点、链码连接前，peer调用func AskAttestation (chaincodeAddr string)(Report, error)功能向指定地址的链码Enclave发送认证请求，并

返回接收的链码认证报告。在链码Enclave接收认证请求后，调用func SendAttestation (peerAddr string, Report struct)(error)功能创建密钥对并读取mrenclave，将公钥和mrenclave发回至peer节点。在peer节点接收链码Enclave发回的信息后，保存链码公钥，调用func VerifyAttestation (IASAddr string, Report struct)(Result, error)功能进行报告验证。func VerifyAttestation (IASAddr string, Report struct)(Result, error)将接收的报告发送给英特尔认证服务（Intel Attestation Service，IAS）中心，IAS中心对报告进行验证并返回验证结果，若经验证报告是可信的，则peer节点与链码建立gRPC连接，方便后续通信。

表 2 E-Fabric 可信接口

API 名称	功能
func AskAttestation(chaincodeAddr string) (Report, error)	向指定地址的链码发送认证请求
func SendAttestation(peerAddr string, Report struct)(error)	将身份认证报告发送给指定地址 peer 节点
func VerifyAttestation(IASAddr string, Report struct)(Result,error)	peer 节点向 IAS 中心发送验证请求
func EncryptOperation(operation struct, key string)(enOperation,error)	加密操作指令（Invoke 或 Query）
func DecryptOperation(enOperation struct, key string)(deOperation,error)	解密操作指令
func SignResult(result struct, key string) (siResult,error)	签名操作指令

在peer节点与链码通信阶段，E-Fabric使用3个API进行通信。当事务到来时，peer节点调用func EncryptOperation (operation struct, key string)(enOperation, error)功能，使用链码公钥对操作指令（Invoke或Query）及其他参数进行加密，并返回加密后的操作指令和参数给链码。链码收到操作指令后，调用func DecryptOperation (enOperation struct, key string)(deOperation, error)功能，使用私钥进行解密，从而得到解密后的操作指令及相关参数。当链码运行结束时，调用func SignResult (result struct, key string)(siResult, error)功能使用私钥对运行结果、读写集等进行签名，防止传输过程中数据被篡改。

2.4 远程认证协议

当链码在Enclave中安全启动后，peer节点需要验证其身份并确认是否可信。为了防止不可信链码连接peer节点，E-Fabric使用远程认证协议验证链码是否在可信Enclave中运行，如果验证失败，则拒绝链码连接请求。

节点与链码Enclave的远程认证协议如图5所示。在链码Enclave创建成功后，首先同步生成非对称密钥对(SK_{CC},PK_{CC})，peer节点将公钥作为链码Enclave的唯一标识。然后链码与指定地址的peer节点建立gRPC连接，peer节点向链码Enclave发送认证请求。当接收认证请求后，链码Enclave生成一个认证报告，该报告可以证明链码身份并表明链码安全运行在Enclave中，报告包含链码Enclave的$mrenclave_{CC}$和公钥PK_{CC}的哈希值。最后链码Enclave将报告和公钥PK_{CC}一同发回至peer节点。

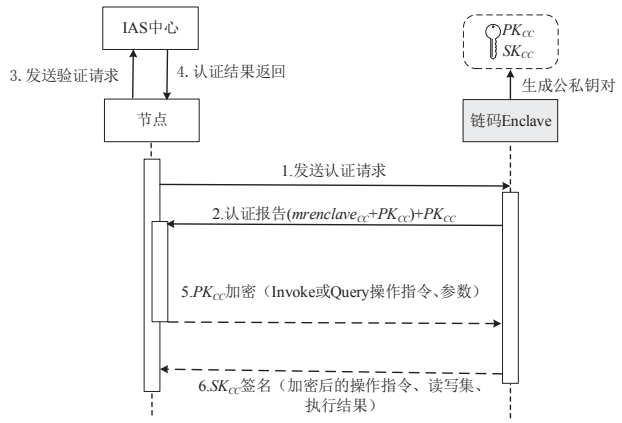

图 5 节点与链码 Enclave 的远程认证协议

peer节点先将报告发送给IAS中心进行验证，然后IAS中心将认证结果发回peer节点。如果验证通过，则peer节点同意链码的连接请求，并保存链码公钥PK_{CC}。当事务到来时，peer节点使用公钥PK_{CC}加密操作指令（Invoke或Query）和参数，并将加密后的数据发送至链码。当链码接收请求参数后，首先依据指令和参数等运行并产生执行结果，然后使用私钥SK_{CC}对执行结果进行签名并发回peer节点，返回结果包括加密后的操作指令、读写集和执行结果。最后对peer节点与链码的通信过程进行加密，以保证通信的安全性。

3 评估测试

3.1 实验环境

E-Fabric基于Hyperledger Fabric v2.3开发，使用EGo v0.4.0、SGX DCAP v1.41驱动程序和efabric/base:0.1镜像构建。

主机设备使用Intel Core（TM）i7-10700 CPU，具有8个内核，支持SGX功能，整个网络在Ubuntu 20.04上搭建并测试。E-Fabric测试网络使用两个组织分别模拟区块链网络中的两家企业或机构，共4个peer节点，2个orderer节点，所有客户端和节点均部署在同一个主机上。

3.2 安全性分析

本节主要分析当前Fabric遭受的攻击和隐私泄露风险，并介绍E-Fabric应对这些攻击的防御情况。

1）镜像替换攻击

Fabric容器创建时，依靠镜像标签为链码创造运行环境。镜像替换攻击是指由于镜像存储在本地，攻击者可以使用带后门的恶意镜像替换原镜像，并使用原Fabric的标签，当程序在后门容器中启动后，攻击者可以进一步窃取运行数据，干扰链码执行。针对此攻击，E-Fabric使用远程认证进行防御。恶意容器中的链码无法创造Enclave，也无法向peer节点提供证明报告，因此peer节点会主动与链码断开连接。只有远程认证成功，peer节点才能与链码正常通信。

2）中间人攻击

Fabric将节点、链码、分发服务放置在独立的Docker容器中运行，程序之间跨容器进行交互。恶意用户通过窃听网络通信获取操作指令以及隐私数据，并对数据进行篡改，将错误的信息传入网络，实施中间人攻击。针对此攻击，E-Fabric在远程认证过程中添加了密钥交换流程，对peer节点和链码通信过程进行加密，这样即使攻击者获取了密文也无法解密。

3）隐私窃取攻击

隐私窃取攻击是指在攻击者侵入Fabric容器的场景下，原Fabric中链码运行不受保护，攻击者可以在容器中窃取运行时的机密数据。针对此攻击，E-Fabric为链码创建了Enclave，Enclave可以保证链码的执行不受同一主机上的攻击者的影响，保护链码运行时的隐私信息和执行结果。

3.3 性能评估

本文使用Hyperledger Caliper[31]测试E-Fabric网络。首先，本文选择Hyperledger Fabric样本中的高吞吐率链码测试用户数量从1增加至128过程中的延迟和吞吐率变化情况，然后将此变化情况与原始Fabric网络进行比较。为了排除网络波动的影响，每轮测试5组数据，并取平均值作为最终数据。读取账本信息（Fabric Query操作）不会创建新的区块或背书，其直接返回读取结果，具有延迟低、效率高等特点；写入账本信息（Fabric Invoke操作）需要背书、分发、验证，其开销较大。这两种操作在性能上有很大差异，因此本文针对这两种情况分别进行测试。

网络延迟与客户端数量的关系如图6所示，吞吐率与客户端数量的关系如图7所示，其中吞吐率指每秒交易的个数。

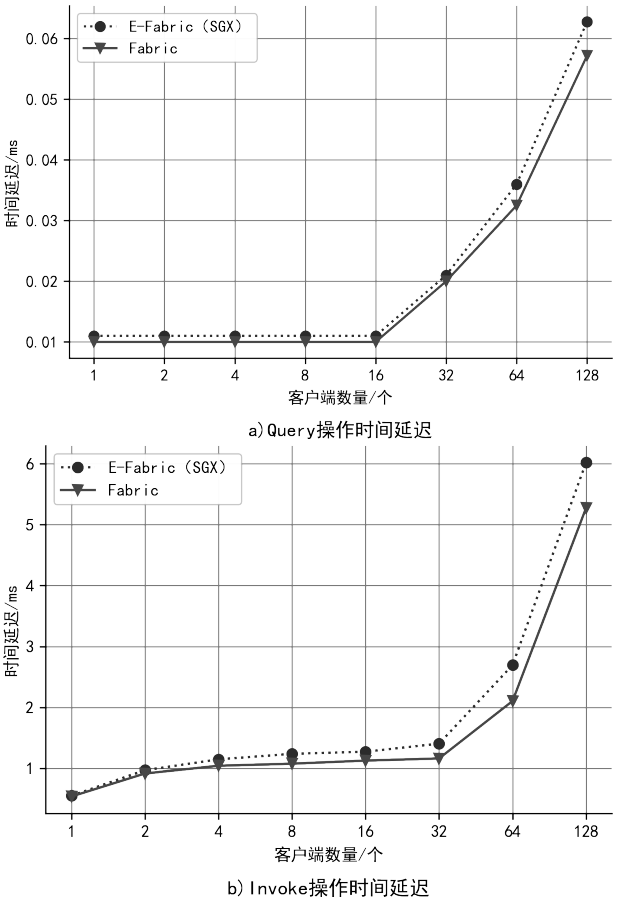

图 6 网络延迟与客户端数量的关系

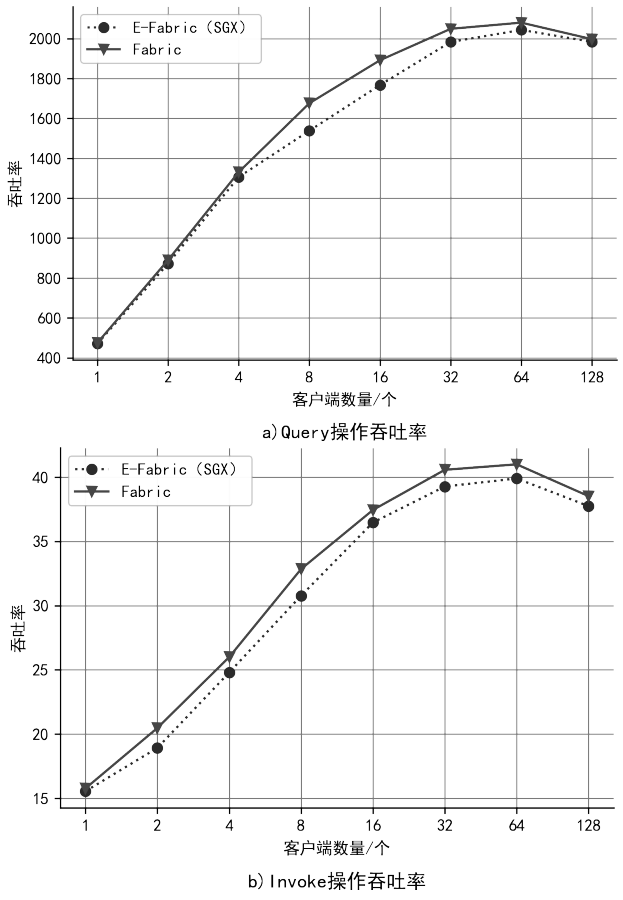

图 7 吞吐率与客户端数量的关系

从图6和图7可以看出，在读取账本和写入账本两种操作的情况下，E-Fabric的延迟略高于原始Fabric网络，其吞吐率略低于原始Fabric网络。考虑Enclave创建、运行都需要额外开销，并且E-Fabric中加入了peer节点与链码的远程认证过程，两者信息交互均需要加/解密，这些都需要额外开销，因此E-Fabric性能适当下降是正常的。对比图6和图7可知，E-Fabric的延迟升高了8%左右，吞吐率下降了4%左右，整体性能达到原始Fabric性能的94%，在可接受范围内。

FPC与E-Fabric性能对比如表3所示，与FPC相比，E-Fabric更加轻量，具有更小的TCB。FPC使用Go语言和C语言重构整个Fabric架构，需要创建分类账本Enclave和链码Enclave，系统由5000行左右可信的C/C++代码组成，其中账本Enclave约有3800行代码，链码Enclave约有1200行代码[13]。除了系统代码以外，FPC还需要将用户编写的链码加载入Enclave中，代码行数为200~5000行，因此可信代码总行数为5000~10000行。根据文献[13]中的测试结果可知，FPC的整体性能达到原Fabric的90%。与FPC相比，E-Fabric采用一个链码Enclave，并且在运行时进行扩展，不需要在系统中增添可信代码。而Enclave只加载用户编写的Go语言链码，代码行数为200~5000行，因此具有更小的TCB。E-Fabric只采用一个Enclave，整体性能更好，可以达到原Fabric的94%。

表 3 FPC 与 E-Fabric 性能对比

区块链名称	Enclave 数量 / 个	可信代码数量 / 行	性能
FPC	2	5000~10000	原 Fabric 的 90%
E-Fabric	1	200~5000	原 Fabric 的 94%

4 相关工作对比

目前，已经有许多工作通过结合SGX和区块链来保证区块链内部隐私安全[10-13,28,32,33]。2019年，CHENG[12]等人提出Ekiden，这是一个将区块链与可信执行环境相结合的系统，该系统将共识与执行分开，可以识别和处理执行过程中出现的问题。2016年，HEARN[33]等人提出Corda项目，该项目用于记录和管理财务协议，将SGX用于某些交易验证，并且在不受信任的节点上运行。Corda在Enclave中执行交易验证，与交易相关的身份信息可以在区块链上加密，并且只在Enclave验证时才会显示身份信息。2019年，RUSSINOVICH[11]等人提出Coco架构，该架构将所有对等节点放入SGX Enclave中，并且仅在受信任的Enclave之间进行通信，导致Enclave中数据过多、TCB过大，对整体性能产生影响。FPC创建了两个Enclave，分别是分类账本Enclave和链码Enclave，并添加了用于远程证明的Enclave注册机构[13]。但是FPC为了适配SGX的分开编写编译机制，使用Go语言和C语言重构了整个Fabric架构，与原架构有较大差异。同时链码只能用C语言编写，目前不支持Go语言，因此不方便用户使用。

现有工作大多针对公有链智能合约安全进行研究，针对私有链Hyperledger Fabric链码安全的研究较少，同时现有研究未对Fabric容器运行环境进行保护，缺少

轻量、安全可信的链码运行环境。目前已有的保护措施对原区块链架构改动较大，引入SGX技术对智能合约编程产生较大更改，不利于用户使用。为了解决上述问题，本文提出E-Fabric架构，针对Fabric容器运行环境构建轻量高效的可信环境，并支持Go语言链码。E-Fabric架构可以在保证链码执行安全性的同时，极大地方便用户使用[34-36]。

5 结束语

区块链系统是公开透明的，其中智能合约的执行不受保护，因此容易泄露用户隐私信息。针对此情况，本文对当前区块链的隐私问题进行分析，提出一种新的隐私保护架构E-Fabric。E-Fabric在独立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，并使用远程认证功能确认链码Enclave身份。本文使用Hyperledger Caliper对E-Fabric进行性能测试，测试结果表明，E-Fabric的性能可以达到原Fabric性能的94%，并且用户在使用E-Fabrc时不需要更改链码编写语言。

参考文献：

[1] NOFER M, GOMBER P, HINZ O, et al. Blockchain[J]. Business & Information Systems Engineering, 2017, 59(3): 183−187.

[2] ZHENG Zibin, XIE Shaoan, DAI Hongning, et al. Blockchain Challenges and Opportunities: A Survey[J]. International Journal of Web and Grid Services, 2018, 14(4): 352−375.

[3] YAGA D, MELL P, ROBY N, et al. Blockchain Technology Overview[EB/OL]. (2019−06−26)[2021−10−13]. https://arxiv.org/abs/1906.11078.

[4] YAJI S, BANGERA K, NEELIMA B. Privacy Preserving in Blockchain Based on Partial Homomorphic Encryption System for AI Applications[C]// IEEE. 2018 IEEE 25th International Conference on High Performance Computing Workshops (HiPCW). New York: IEEE, 2018: 81−85.

[5] SUN Xiaoqiang, YU F R, ZHANG Peng, et al. A Survey on Zero−Knowledge Proof in Blockchain[J]. IEEE Network, 2021, 35(4): 198−205.

[6] YE Shaochen. Research on Verifiable Byzantine Fault Tolerant Consensus Mechanism Based on Secure Multi−Party Computing in Blockchain[D]. Beijing: Beijing University of Posts and Telecommunications, 2021.
叶少宸 . 区块链中基于安全多方计算的可验证拜占庭容错共识机制的研究 [D]. 北京：北京邮电大学，2021.

[7] WANG Cuihan, ZHAO Chen, XU Xiaogang, et al. Distributed Paralel Computing Environment: MPI[J]. Computer Science, 2003, 30(1): 25−26.
王萃寒，赵晨，许小刚，等 . 分布式并行计算环境：MPI[J]. 计算机科学，2003，30（1）：25−26.

[8] COSTAN V, DEVADAS S. Intel SGX Explained[EB/OL]. (2016−08−06)[2021−09−13]. http://css.csail.mit.edu/6.858/2020/readings/costan−sgx.pdf.

[9] Intel Corporation. Intel(R) Software Guard Extensions Developer Guide[EB/OL]. (2021−11−02)[2021−12−20]. https://download.01. org/intel−sgx/sgx−linux/2.15.1/docs/Intel_SGX_Developer_Guide.pdf.

[10] KOSBA A, MILLER A, SHI E, et al. Hawk: The Blockchain Model of Cryptography and Privacy−Preserving Smart Contracts[C]//IEEE. 2016 IEEE Symposium on Security and Privacy (SP). New York: IEEE, 2016: 839−858.

[11] RUSSINOVICH M, ASHTON E, AVANESSIANS C, et al. CCF: A Framework for Building Confidential Verifiable Replicated Services[EB/OL]. (2019−09−24)[2021−12−03]. https://www.microsoft.com/en−us/research/publication/ccf−a−framework−for−building−confidential−verifiable−replicated−services/.

[12] CHENG R, ZHANG Fan, KOS J, et al. Ekiden: A Platform for Confidentiality−Preserving, Trustworthy, and Performant Smart Contracts[C]// IEEE. 2019 IEEE European Symposium on Security and Privacy (EuroS&P). New York: IEEE, 2019: 185−200.

[13] BRANDENBURGER M, CACHIN C, KAPITZA R, et al. Trusted Computing Meets Blockchain: Rollback Attacks and a Solution for Hyperledger Fabric[C]//IEEE. 2019 38th Symposium on Reliable Distributed Systems (SRDS). New York: IEEE, 2019: 324−329.

[14] ANDROULAKI E, BARGER A, BORTNIKOV V, et al. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains[C]// ACM. Proceedings of the Thirteenth EuroSys Conference. New York: ACM, 2018: 1−15.

[15] CACHIN C. Architecture of the Hyperledger Blockchain Fabric[EB/OL]. (2016−06−04)[2021−12−26]. https://www.zurich.ibm.com/dccl/papers/cachin_dccl.pdf.

[16] CUI Pinchen, UMPHRESS D. Perturbing Smart Contract Execution through the Underlying Runtime[C]//Springer. International Conference on Security and Privacy in Communication Systems. Heidelberg: Springer, 2020: 336−349.

[17] Edgeless Systems GmbH. Build Confidential Go Apps with Ease[EB/OL]. (2021−10−06)[2021−12−20]. https://www.ego.dev.

[18] WANG Juan, FAN Chengyang, CHENG Yueqiang, et al. Analysis and Research on SGX Technology[J]. Journal of Software, 2018, 29(9): 2778−2798.
王鹃，樊成阳，程越强，等 . SGX 技术的分析和研究 [J]. 软件学报，2018，29（9）：2778−2798.

[19] Intel Corporation. Introduction to Intel® SGX Sealing[EB/OL]. (2016−05−04)[2021−11−20]. https://www.intel.com/content/www/us/en/developer/articles/technical/introduction−to−intel−sgx−sealing.html.

[20] Intel Corporation. Innovative Technology for CPU Based Attestation and Sealing[EB/OL]. (2013−08−14)[2021−11−20]. https://www.intel.com/content/www/us/en/developer/articles/technical/innovative−technology−for−cpu−based−attestation−and−sealing.html.

[21] SSLab. SGX101 Attestation[EB/OL]. (2019−11−17)[2021−11−20]. https://sgx101. gitbook.io/sgx101/sgx−bootstrap/attestation.

[22] ZHANG Rui, XUE Rui, LIU Ling. Security and Privacy on Blockchain[J].

ACM Computing Surveys (CSUR), 2019, 52(3): 1-34.
[23] MEIKLEJOHN S, POMAROLE M, JORDAN G, et al. A Fistful of Bitcoins: Characterizing Payments among Men with No Names[C]//ACM. 2013 Conference on Internet Measurement Conference. New York: ACM, 2013: 127-140.
[24] MÖSER M, BÖHME R. The Price of Anonymity: Empirical Evidence from a Market for Bitcoin Anonymization[J]. Journal of Cybersecurity, 2017, 3(2): 127-135.
[25] REID F, HARRIGAN M. An Analysis of Anonymity in the Bitcoin System[C]//Springer. Security and Privacy in Social Networks. Heidelberg: Springer, 2013: 197-223.
[26] RON D, SHAMIR A. Quantitative Analysis of the Full Bitcoin Transaction Graph[C]//Springer. International Conference on Financial Cryptography and Data Security. Heidelberg: Springer, 2013: 6-24.
[27] IBM. MSP Implementation with Identity Mixer[EB/OL]. (2017-11-23)[2021-11-21]. https://hyperledger-fabric.readthedocs.io/en/release-1.2/idemix.html.
[28] YUAN Rui, XIA Yubin, CHEN Haibo, et al. Shadoweth: Private Smart Contract on Public Blockchain[J]. Journal of Computer Science and Technology, 2018, 33(3): 542-556.
[29] Docker. Base Image for Applications Using the Official Intel SGX SDK[EB/OL]. (2021-02-11)[2021-12-22]. https://registry.hub.docker.com/r/ffosilva/sgx.
[30] Felix Schuster. EGo: Effortlessly Build Confidential Apps in Go[EB/OL]. (2021-02-21)[2021-12-22]. https://blog.edgeless.systems/ego-effortlessly-build-confidential-apps-in-go-dc2b1460e1bf.
[31] IBM. Hyperledger Caliper Documentation[EB/OL]. (2021-04-16)[2021-12-15]. https://hyperledger.github.io/caliper/v0.4.2/getting-started/.
[32] CAI Chengjun, XU Lei, ZHOU Anxin, et al. EncELC: Hardening and Enriching Ethereum Light Clients with Trusted Enclaves[C]//IEEE. IEEE INFOCOM 2020-IEEE Conference on Computer Communications. New York: IEEE, 2020: 1887-1896.
[33] HEARN M, BROWN R G. Corda: A Distributed Ledger[J]. (2016-11-29)[2021-10-11]. https://www.corda.net/content/corda-technical-whitepaper.pdf.
[34] ZHONG Yuhan. Design and Development of Trusted Computing Platform Based on Blockchain and TEE[D]. Hangzhou: Zhejiang University, 2021.
钟雨涵 . 基于区块链和 TEE 的可信计算平台设计与开发 [D]. 杭州：浙江大学，2021.
[35] ZHANG Xuewang, YIN Zijie, FENG Jiaqi, et al. Data Trading Scheme Based on Blockchain and Trusted Computing[J]. Journal of Computer Applications, 2021, 41(4): 939-944.
张学旺，殷梓杰，冯家琦，等 . 基于区块链与可信计算的数据交易方案 [J]. 计算机应用，2021，41（4）：939-944.
[36] LIU Qianren, XUE Miao, REN Mengxuan, et al. Application and Research of Blockchain in Data Sharing and Trusted Computing[J]. Designing Techniques of Posts and Telecommunications, 2020(11): 18-23.
刘千仞，薛淼，任梦璇，等 . 基于区块链的数据共享与可信计算应用与研究 [J]. 邮电设计技术，2020（11）：18-23.