一种基于以太坊字节码的隐蔽通信方法

黄冬艳[1,2]，黄珉[1,2]
（1.桂林电子科技大学信息与通信学院,广西 桂林 541000；2.认知无线电与信息处理省部共建教育部重点实验室,广西 桂林 541000）

摘 要：针对通信场景中对信息安全性要求极高、数据泄露可能引发重大风险的领域，提出一种利用以太坊虚拟机字节码的隐蔽通信方案，可实现敏感信息的隐蔽传输。该方案通过对智能合约变量存储空间的策略性分配，将隐蔽数据嵌入合约字节码中，通过字节码本身特性设置定位标识，使接收方能高效提取隐藏信息。此外，方案提出三种密文解析模式以支持不同规模密文的传输，进一步增强了编码数据的安全性。理论分析与大量实验结果表明，该方案单笔交易可有效隐藏多达170比特信息，嵌入合约与原始合约的操作码频率分布结构相似度最高可达99.78%。正常合约与嵌入合约的高频3-gram操作码模式的皮尔逊相关系数达0.9997（$p = 6.42\times10^{-14}$），表明嵌入过程未在局部指令序列分布中引入统计学显著差异。这些结果充分证明了所提方案具有强大的隐蔽能力、传输效率与安全性。

关键词：区块链；以太坊；隐蔽通信；字节码；智能合约
中图分类号：TP393.0
文献标志码：A
DOI：10.11959/j.issn.1000–436x

A Covert Communication Method Based on Ethereum Bytecode

Huang Dongyan[1,2], Huang Min[1,2]
1. School of Information and Communication Technology, Guilin University of Electronic Technology, Guilin 541000, China
2. Ministry of Education Key Lab. of Cognitive Radio and Information Processing, Guilin 541000, China

Abstract: This paper presents a covert communication scheme leveraging the Ethereum Virtual Machine (EVM) bytecode to enable the undetected transmission of sensitive information. The scheme embeds covert data into the bytecode of smart contracts by strategically allocating storage space for contract variables, with locators embedded within the bytecode to facilitate efficient extraction of the hidden messages by the intended recipient. Additionally, three ciphertext parsing modes are proposed to support the transmission of ciphertexts of varying sizes, enhancing the security of the encoded data. Both theoretical analysis and extensive experimental results demonstrate that the scheme can effectively conceal up to 170 bits of information per transaction, achieving a high degree of structural similarity—up to 99.78%—between the opcode frequency distributions of the embedded and original contracts. Notably, the Pearson correlation coefficient between the top-10 3-gram opcode patterns in normal and embedded contracts is 0.9997 (p-value = 6.42×10^{-14}), indicating that the embedding process introduces no statistically significant deviations in the local instruction sequence distributions. These results highlight the proposed scheme's strong concealment capabilities, transmission efficiency, and security.

Keywords: Blockchain, Ethereum, covert communication, bytecode, smart contract

收稿日期：XXXX–XX–XX；修回日期：XXXX–XX–XX
通信作者：黄冬艳，huangdongyan-gua@163.com
基金项目：广西自然科学基金面上项目（2025GXNSFAA069685）
Foundation Items：Guangxi Natural Science Foundation General Project (Grant 2025GXNSFAA069685).

1 引言

隐蔽通信通过难以被察觉的方式传输数据，以增强信息传递过程的安全性，在需避免通信行为暴露的场景中具有重要应用价值[1]。在军事行动、外交谈判及反商业间谍等对信息安全性要求极高、数据泄露可能引发重大风险的领域中，此类通信方式发挥着不可替代的关键作用[2]。在信息隐蔽性要求极高的应用场景中，传统隐蔽通信方法仍面临多重挑战：其消息往往具有明确的指向特征，易被识别与追踪；依赖可检测的传输信道，增加了被拦截的风险；对集中式攻击的脆弱性可能扩散至整个通信系统；此外，信息隐蔽程度不足也可能直接导致敏感内容泄露。这些结构性缺陷使得传统方法在对抗性环境中容易成为监测与破解的重点目标[3]。区块链技术凭借其去中心化、分布式账本存储、防篡改特性、用户伪匿名性及广播通信能力等内在属性，与隐蔽通信的需求高度契合，成为实施隐蔽通信系统的理想平台[4]。

以太坊[5]作为最具影响力的公有区块链之一，是区块链2.0时代的标志性平台。其核心创新在于支持可编程智能合约，使开发者能够定义、部署并自动化执行去中心化应用逻辑。智能合约[6]是一种自动执行、强制执行和记录预定义规则的计算机程序。这些合约通常使用Solidity等高级语言编写，被编译为字节码后由以太坊虚拟机(EVM)执行。编译后的字节码由一系列操作码构成，每个操作码对应EVM中的一项底层操作。这种机器级的表示形式掌控计算、内存访问与存储操作，构成了已部署合约的核心逻辑。值得注意的是，字节码的结构特性与不可直接阅读特性使其成为承载隐蔽信息的理想载体。

与传统依赖多媒体载体（如图像或音频）的隐写方法不同，以太坊提供了一种自成体系且去中心化的通信通道。字节码作为原生执行格式，无需依赖外部数据分发，更不易被篡改或过滤。此外，字节码包含了初始化组件与运行时组件，其可观的体积与复杂度为隐蔽数据提供了天然掩护。其二进制形式在提升效率的同时，也使得嵌入内容更难以通过人工检查或自动化分析被察觉。这些特性共同使以太坊字节码成为隐蔽通信的有效载体。

然而，基于字节码的隐蔽通信方案面临三个关键挑战：

首先，如何在确保合约正常执行的前提下嵌入隐蔽信息？合约中添加额外代码可能引发非预期错误，生成合约时必须遵循严格的编码规范。其次，如何保证接收方能高效且准确地提取隐蔽信息？Solidity字节码作为EVM执行的二进制指令集，具有高度抽象与底层细节并存的特征，难以直接解读。此外，字节码通常包含大量冗余部分，若未掌握嵌入规则，接收方将难以定位有效信息。第三，如何有效控制合约执行成本？大量增加的字节码会导致部署成本攀升与合约复杂度增加。因此，在实现隐蔽通信的同时，需对字节码规模进行精细管控，以确保执行效率和成本节约。

本文提出一种基于以太坊字节码的隐蔽通信方案。所提方案利用PUSH1指令的使用频次及其所携带的8位操作数值这两个维度，对隐蔽信息进行编码。同时，方案采用require函数标记隐蔽通信段，使接收方能够从难以直接解读的字节码中准确提取隐蔽信息。本研究的贡献可概括如下：

1)提出一种创新型隐蔽通信方案，通过控制PUSH1指令的使用次数与PUSH1指令携带的8位操作数值，将密文以字节码的形式编码，同时保持合约功能完整性。 2)设计完整的嵌入与解码框架，包含三种密文解析模式（EQ、GT、LT）及基于错误信息哈希的定位机制，实现隐蔽信息的高效精确提取。 3)基于以太坊浏览器Etherscan提供的70,000余个真实智能合约开展大规模实验，证明本方案具备高传输容量（单笔交易最高170比特）、强结构隐蔽性（Kullback-Leibler散度0.0008）、以及与传统安全工具（Slither）的良好兼容性。特别值得注意的是，正常合约与嵌入合约的高频3-gram操作码模式皮尔逊相关系数达到0.9997（对应p值为6.42×10^{-14}），进一步证实嵌入过程不会在局部指令序列分布中引入统计学显著差异。

本文余下部分组织结构如下：第二节探讨区块链与以太坊中实现隐蔽通信的相关工作，并对现有研究成果进行总结；第三节介绍基于字节码的隐蔽通信设计方案，阐述字节码隐写的基本原理；第四节详细阐述隐蔽通信的具体实施流程；第五节对提出方案的性能表现进行全面分析；最后第六节总结全文。

2　相关工作

当前基于区块链的隐蔽通信研究可分为三类：存储型隐蔽信道、时间型隐蔽信道以及复合型隐蔽信道。

存储型隐蔽信道利用区块链上的交易数据或区块数据隐藏信息。例如，Partala[7]将加密信息嵌入支付地址的最低有效位，并控制相关交易在统计特性上与正常交易无异，实现了区块链上的隐蔽通信。

时间型隐蔽信道通过利用区块链交易的时间间隔或区块生成的时间特性传递信息。黄冬艳等人[8]提出一种多地址时间型隐蔽通信方法，以解决现有基于时间戳的隐藏方法信息容量低的问题。

复合型隐蔽信道通常融合多种方法构建隐蔽通信系统，基于智能合约的隐蔽信道[9]就是其中之一。智能合约参数的多样性、数据冗余性以及代码可编程性使其成为构建隐蔽信道的优良载体。例如，Zhang 等人[10]提出一种融合智能合约的隐蔽通信模型，该模型将秘密信息序列映射至合约参数，通过调用合约实现信息传输。

此外，现有研究也存在将隐写术与以太坊相结合的工作。Zhang 等人[11]将密文编码至投票与投标合约的结构中，辅以加密提交与两阶段协议，以缓解区块链透明性带来的隐私风险。刘九良[12]探索了在不改变合约语义的条件下，通过重排源代码与字节码结构来隐藏信息的方法。Liu 等人[13]提出一种相关方案，通过使用 HMAC 调制交易 VALUE 字段实现多比特嵌入，在保证载体合理使用的同时提升了隐蔽信息容量。

除合约层面与交易信道外，Zhang 等人[14]探究了利用 Whisper 协议作为隐写传输媒介的方法。Basuki 与 Rosiyadi[15]提出了一种结合区块链与多媒体域的混合方法，通过引入分组编码与多字段嵌入来共同提升容量并维持行为真实性。与此同时，Gimenez-Aguilar 等人[16]设计了 Zephyrus 系统，将消息嵌入交易元数据与合约状态中，并通过受控实验验证了其可行性。

目前，基于区块链的隐蔽通信研究主要集中在应用层或协议层，多利用区块链的公开数据字段或智能合约参数作为隐蔽信息的载体。例如，Zhang 等人[11]通过智能合约参数映射实现信息隐藏；佘维等人[17]提出了基于智能合约操作码序列重构的隐蔽通信模型，但其核心仍是通过操作码序列的统计特征进行信息隐藏，并未触及 EVM 指令层的直接修改；此外，佘维等人[18]利用智能合约调用参数作为隐蔽载体，本质上仍属于应用层隐写范畴。

在现有研究中，仅有少数工作尝试在字节码层面实现隐写。例如，刘九良[12]提出“跨基本块”重新设计控制流路径的方法，但此类修改常与 Solidity 优化器产生冲突，可能导致不可预测的漏洞。Zhang 等人[11]的研究因其多轮交互特性易产生延迟与过大的成本开销。此外，Zephyrus 系统[16]提及了字节码级嵌入的潜力，但未提供具体实施方案。

值得一提的是，近年来生成式隐写作为信息隐藏领域的重要分支，也取得了显著进展。生成式隐写以秘密信息为驱动直接生成含密载体，避免了对现有载体的修改，从而具备较强的抗隐写分析能力。周志立等人[19]对生成式隐写进行了系统综述，将其分为图像、文本、音频和社交网络行为四类，并总结了各类方法的优缺点及未来发展方向。此外，周志立等人[20]提出了一种基于轮廓自动生成的构造式图像隐写方法，通过 LSTM 生成轮廓并利用 pix2pix 模型生成含密图像，实现了较高的隐蔽容量和抗隐写分析性能。

总体而言，上述研究极大地推动了区块链隐蔽通信的发展，但大多数工作仍停留在应用层或协议层，利用公开数据字段或合约参数作为载体，并未深入到由编译器生成的 EVM 字节码指令序列本身进行编码，区块链隐蔽通信方法对比如表 1 所示。与之不同，本研究聚焦于字节码层面，嵌入过程直接与 EVM 指令及存储布局交互，通过利用编译器输出的确定性特征，构建了一种结构集成的隐蔽信道。关键在于，本方案通过在编译时策略性地安排变量存储布局，将密文编码于字节码序列中，而无需修改合约的运行时逻辑，从而保持了合约功能的完整性与执行语义的不变。本文工作表明，此前尚未被充分探索的字节码层能够支持语义保持的隐蔽通信，为去中心化系统中的安全信息传递提供了新的研究方向与技术路径。

3　基于字节码的隐蔽通信设计

3.1　隐蔽通信模型

如图 1 所示，在隐蔽通信场景中，*Alice* 与 *Bob*

表1　区块链隐蔽通信方法对比

文献	隐蔽信息载体	载体性质	交互轮次	成本特征
Partala [7]	交易地址最低有效位	应用层数据字段	单轮	低
黄冬艳等[8]	交易时间戳	协议层时间特性	多轮	中
Zhang 等[11]	智能合约参数	应用层合约参数	多轮	高
Liu 等[13]	交易 VALUE 字段	应用层数据字段	单轮	中
Gimenez-Aguilar[16]	交易元数据与合约状态	应用层数据字段与合约状态	多轮	中
刘九良[12]	字节码重排	字节码层	单轮	中
余维等[17]	操作码统计特征	应用层统计特性	单轮	中
本文	EVM 字节码	字节码层	单轮	低

分别扮演发送方与接收方角色。他们通过公共以太坊区块链信道传输信息，并努力避免被对手 *Wendy* 侦测到通信行为。在该场景中，*Alice* 将加密信息嵌入智能合约并通过公共以太坊区块链信道发送；对手 *Wendy* 则试图检测并解密隐蔽信息；另一方面，*Bob* 从以太坊区块链获取智能合约，采用与嵌入方案对应的提取方案获得加密的隐蔽信息，最终使用解密密钥重构原始消息。

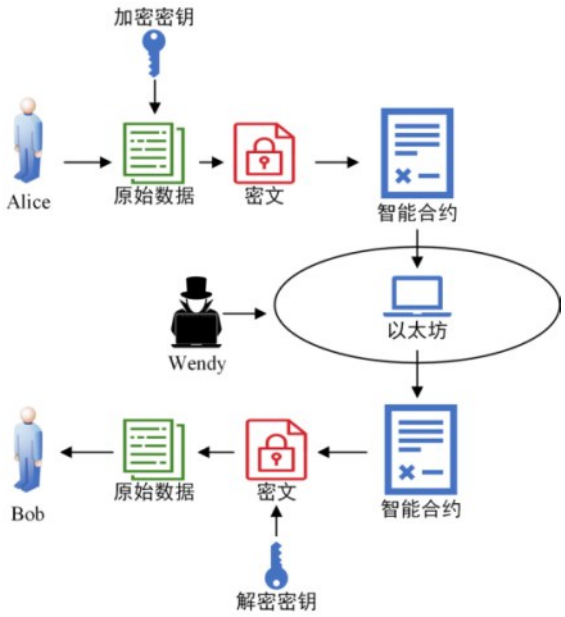


图1　隐蔽通信模型

3.2　隐蔽信息载体

本文实验中的智能合约采用 Solidity 语言编写，并以 Solidity 0.8.0 版本为基础开展。相比前代版本，Solidity 0.8.0 在语言规范与编译器实现上更具代表性，其底层机制也更为稳定。该版本将算术溢出检查及统一的错误处理机制设为默认行为，规范了 EVM 执行行为的标准化。EVM 的核心操作码如表 2 所示。

表2　EVM 核心操作码

栈操作码	PUSH i, DUP j, SWAP j, POP
算术操作码	ADD, MUL, SUB, DIV, MOD
比较操作码	ISZERO, LT, GT, EQ
逻辑操作码	NOT, AND, OR
分支跳转操作码	JUMP, JUMPI, JUMPDEST
内存操作码	MSTORE, MLOAD, MSTORE8
存储操作码	SSTORE, SLOAD

为评估关键操作码在不同 Solidity 版本中的分布稳定性，我们从合约数据库中筛选出相同数量的真实合约对其操作码的使用率进行统计分析。表 3 列出了三种 Solidity 版本中使用率最高的 8 种操作码。数据显示，PUSH1 是所有版本智能合约中最常用的操作码，平均使用率稳定在 10.50% 左右。该指令常用于存储空间与调用栈之间的数据传输，可携带一个 8 位操作数。PUSH1 的高频使用特性可使其作为高效的隐蔽通信载体，其携带的 8 位操作数为隐藏信息的嵌入提供了充足的容量。本文提出的字节码隐写方案，正是利用 PUSH1 指令的使用频率及其携带的 8 位操作数值来实现隐蔽通信。尽管不同版本 Solidity 中操作码的使用率存在一定波动，但测试显示 PUSH1 的使用率始终最高，因此本文提出的隐蔽通信方法具有适用性，且可根据不同 Solidity 版本进行相应调整。

3.3　嵌入方法

在阐述所提隐蔽通信方法的原理之前，首先简要说明 EVM 中的存储分配机制。智能合约的变量

表3　三种Solidity版本中使用率最高的8种操作码

操作码	0.8.0版本平均使用率	0.5.17版本平均使用率	0.8.28版本平均使用率
PUSH1	10.51%	10.58%	10.47%
PUSH2	7.46%	7.42%	7.48%
POP	7.04%	6.98%	7.06%
JUMPDEST	6.99%	6.92%	6.98%
JUMP	5.30%	5.27%	5.25%
SWAP1	4.99%	5.05%	4.93%
DUP2	3.58%	3.55%	3.52%
DUP1	3.52%	3.51%	3.47%

声明后，EVM 会为其分配存储位置。该分配过程涉及大量操作码，其中用于将数据压入栈的 PUSH1 指令起到关键作用。PUSH1 携带的 8 位操作数用于指定变量的预分配存储位置。

为展示如何通过 PUSH1 等特定指令操作嵌入隐蔽信息，现以三个变量的存储分配为例进行说明。假设我们声明三个定长变量：a、b 和 c，其数据类型分别为 uint X、uint Y 与 uint Z。此处 X 表示变量 a 的尺寸为 X/8 字节，Y 和 Z 同理。

合约变量声明后，EVM 从首个存储槽开始分配存储空间。第一个变量 a 将存储于槽 0。由于每个存储槽的最大容量为 32 字节，当 X/8 = 32 时，槽 0 即被占满。此时，EVM 将转向槽 1 存储第二个变量 b。若 Y/8 < 32，则槽 1 的剩余空间为 (32 − Y/8) 字节。

随后，在存储第三个变量 c 之前，EVM 会检查 Z 的大小。若 Z/8 小于槽 1 的剩余空间 (32−Y/8)，c 将被存储于槽 1；若 Z/8 超过槽 1 的可用空间，EVM 则将转移到槽 2 存储 c，而槽 1 的剩余空间将不再被使用。

假设 X、Y、Z 的值分别为 256、128 和 128。根据 EVM 的存储布局规则，变量 a 将独占槽 0。变量 b 和 c 则会被打包存储在同一个槽 1 中。在相应的字节码中，EVM 首先通过 PUSH1 0x00 指令，将槽 0 的标识推入栈中，用于定位变量 a。接着，为存储变量 b，EVM 执行 PUSH1 0x01 指令，表明后续操作将面向槽 1。由于 b 是槽 1 中的第一个变量，其在该槽内的起始偏移量为 0，因此通过另一个 PUSH1 0x00 来指定这一偏移量。对于变量 c，EVM 同样使用 PUSH1 0x01 指令指向槽 1，并通过 PUSH1 0x10（十进制 16）指明 c 在该槽内的起始偏移量，即紧接 b 之后的位置。上述存储操作如图 2 所示。

至此可见，PUSH1 的使用频率及其后续操作数可由变量的存储位置决定。因此，通过控制变量数据类型所占空间的大小，我们可在存储中嵌入密文序列。

3.4　变量选择

为实现本文所提方案，合理的变量选择至关重要。智能合约变量包含状态变量、结构体及映射等多种声明形式，各类声明遵循不同的存储规则，可归纳为定值存储与动态存储两类。对于动态存储变

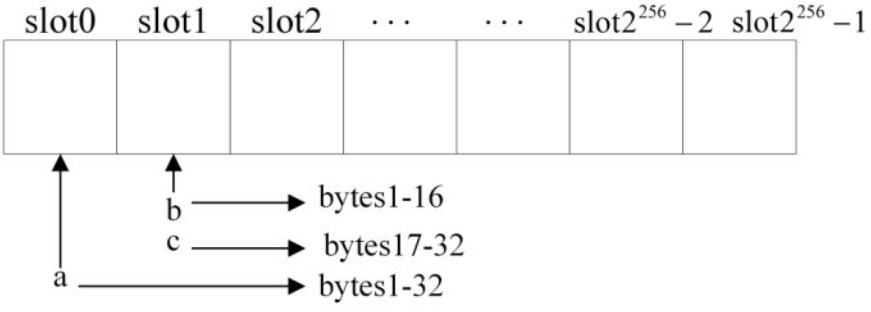


图2　变量存储结构示意图

量，EVM无法预先确定其所需存储空间，因此Solidity会通过哈希函数动态计算其存储位置。该过程涉及操作码的数量庞大，若使用PUSH1指令嵌入隐蔽信息将变得复杂。

相比之下，对于定值存储变量，EVM会为数据分配预设的存储位置。这一过程涉及的操作码更少且更可控，更利于通过PUSH1指令嵌入隐蔽信息。因此，本文选择声明存储定值的公共状态变量与结构体成员变量。需要特别说明的是，EVM会优先为公共状态变量分配存储空间，其次才处理结构体成员变量。这意味着即使结构体成员变量在代码中声明顺序在前，EVM仍会优先为公共状态变量预留存储槽空间。

3.5　隐蔽信息提取

为便于隐蔽信息的精确定位与提取，本文利用require函数对携带隐蔽数据的PUSH1指令范围进行界定。require函数在Solidity中用于验证条件并在条件不满足时回退交易。具体而言，require函数通过变量间的数值比较实现范围划分，其中用于比较的变量对均嵌入了密文。执行过程中，EVM会依次从存储中加载这些变量，从而形成连续编码隐藏信息的字节码区域。该方法支持在单一逻辑块内实现高容量信息嵌入。

在Solidity中，当require语句包含错误信息参数（如require(...,“msg”)）时，编译器会在生成的字节码中固定插入一个对应的32字节错误函数选择器。该选择器具有固定的32字节十六进制表示形式：08C379A0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000，具体字节码片段如图3所示。为简洁起见，本文将前四个字节08C379A0称为函数选择器。该选择器在编译后即稳定存在于合约字节码内，无论require的条件在执行时是否成立。因此，本方案将其作为可靠的定位标记，用于识别嵌入数据段的边界。这一机制不依赖交易执行是否触发回退，信息提取完全基于字节码中的静态特征，与运行时状态无关。

通过检测此模式，接收方能够精准隔离与每个require子句关联的字节码段，进而解码嵌入的密文。此外，require函数通过三种关系运算符确定条件真值，每种运算符对应不同的字节码。本文基于这三种关系运算符的字节码特征，提出三种密文解析模式。接收方将根据接收到的字节码类型选择相应的密文解析方法。

```
60005410610175576040517f08c
379a0000000000000000000000
0000000000000000000000000000
00000008152600401610 16c9061
02dd565b60405180910390fd
```

图3　具体字节码片段中的32字节函数选择器

4　隐蔽通信过程

在开始隐蔽通信前，双方需预先具体约定三种密文解析模式，并确定解析后密文的排序规则以重构完整信息。基于上述理论设计的隐蔽通信流程如图4与图5所示。

信息处理：发送方首先对明文消息进行处理，将其转换为二进制编码格式。

分配存储空间：随后，发送方编写智能合约，声明V个无符号整型变量及其他辅助变量，其中V代表公共状态变量与结构体成员变量的总数。EVM将从存储区域起始处开始为这些变量线性分配存储空间。需注意，此分配过程具有线性与不可逆性，一旦为变量分配了存储空间，则无法修改先前的分配。通过调整变量声明顺序，可使存储位置参数与密文的二进制编码相对应。如图4所示，通过定义四个无符号整型变量对二进制消息"100100"进行编码。编译过程中，Solidity编译器依据变量声明决定存储布局，并将其转换为特定的字节码结构，密文隐含嵌入其中。接着，利用require函数将包含隐藏信息的字节码段连接起来，形成一个可恢复的密文块。

条件检查：发送方为已声明的变量赋值，并在合约中添加require语句。变量按照二进制密文指定的顺序置于各require子句中，以实现条件比较。根据关系运算符定义了三种解析模式：

· 相等运算符对应EQ操作码，密文由PUSH1指令的使用数量表示；

· 大于运算符对应GT操作码，密文由各PUSH1指令携带的8位操作数值编码；

· 小于运算符对应LT操作码，其作为干扰项，不嵌入任何信息。

隐蔽方法涉及的主要操作码及其对应字节码如

表 4 所示。require 子句的执行顺序依据二进制消息进行排列，以确保正确解码。完整编码过程如算法 1 所述。

算法 1

输入： 二进制消息 $M = \{b_1, b_2, ..., b_n\}$

输出： 嵌入了隐蔽信息的智能合约 C

1) 将消息 M 分割为多个段 $S = \{s_1, s_2, ..., s_k\}$

2) 初始化变量列表 $V \leftarrow \varnothing$

3) 初始化语句列表 $R \leftarrow \varnothing$

4) for 每个消息段 $s_i \in S$ do

5) if 为 s_i 选择 GT 模式 then

6) 将 s_i 编码为 PUSH1 指令携带的操作数值

7) 创建用于比较的变量对 (v_i, v_j)

8) 将 v_i, v_j 追加到 V

9) $R \leftarrow R \cup \{require(v_i > v_j, "GT")\}$

10) else if 为 s_i 选择 EQ 模式 then

11) 通过重复 PUSH1 指令编码 s_i（重复次数 = 比特长度）

12) 创建用于相等比较的变量对 (v_i, v_j)

13) 将 v_i, v_j 追加到 V

14) $R \leftarrow R \cup \{require(v_i == v_j, "EQ")\}$

15) else 使用 LT 模式作为混淆

16) 生成虚拟比较对 (v_i, v_j)

17) 将 v_i, v_j 追加到 V

18) $R \leftarrow R \cup \{require(v_i < v_j, "LT")\}$

19) end if

20) end for

21) 使用变量列表 V 和语句列表 R 构建 Solidity 合约 C

22) return C

表 4　隐蔽方法涉及的主要操作码及其对应字节码

方法中涉及的操作码	对应字节码
PUSH1	0x60
LT	0x10
GT	0x11
EQ	0x14

信息恢复：合约部署完成后，将在区块链上发起交易。接收方查询已部署的合约，并从交易信息中获取其字节码。随后，接收方遍历字节码，定位标志 require 函数所生成错误处理区域的 08C379A0

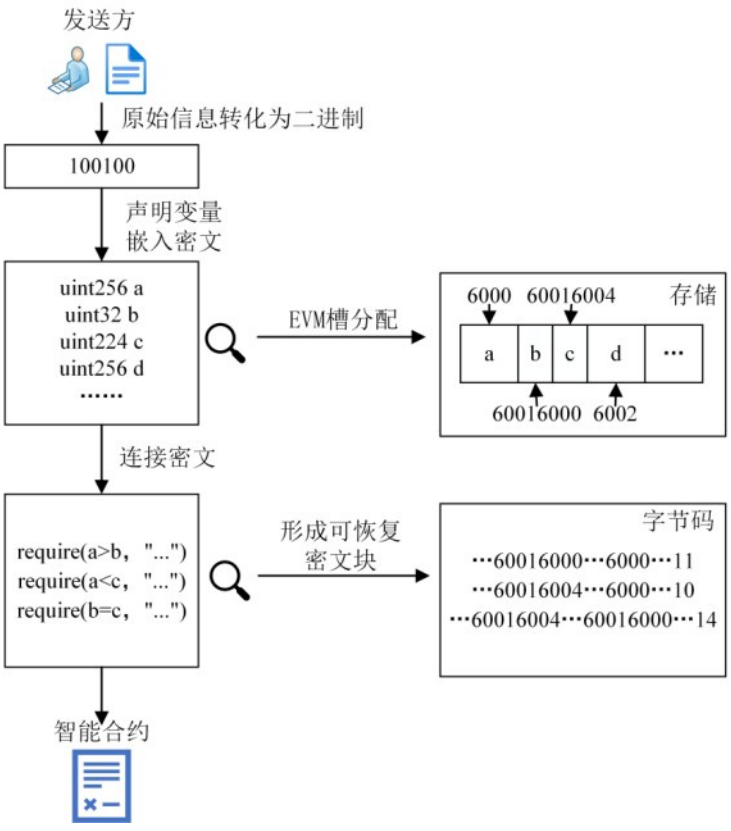


图 4　发送方加密过程

函数选择器，从而确定嵌入密文的范围。在每个此类区域内，识别对应的关系运算符，并基于检测到的运算符确定密文解析模式如下：

· 在 EQ 模式下，将该区域内字节码 0x60（PUSH1）的出现次数转换为二进制，以表示密文；

· 在 GT 模式下，提取每个 PUSH1 字节码后携带的 8 位操作数，将其转换为二进制并顺序拼接；

· 在 LT 模式下，该区域不含有效密文，直接跳过。

如图 5 所示，接收方通过定位 08C379A0 的出现位置，精准识别出包含嵌入数据的字节码段。随后，根据相应的解析模式提取二进制密文，并按照预定义的顺序重新组装，以恢复原始信息。解码过程的正式描述如算法 2 所示。

算法 2

输入：智能合约字节码 B

输出：解码后的二进制消息 M

1) 初始化 M ← 空字符串

2) 识别字节码 B 中包含函数选择器 08C379A0 的所有字节码段

3) for 每个定位到的字节码段 seg ∈ B do

4) 确定 seg 中的比较操作符操作码

5) if 操作码 = 0x11 (GT) then

6) 从 PUSH1 指令中提取操作数值

7) 将操作数转换为二进制 → 比特串

8) 将比特串追加到 M

9) else if 操作码 = 0x14 (EQ) then

10) 统计 PUSH1 操作码的数量

11) 将数量转换为二进制 → 比特串

12) 将比特串追加到 M

13) else if 操作码 = 0x10 (LT) then // 这是虚拟段，忽略

14) continue

15) end if

16) end for

17) return M

5　隐蔽通信系统性能分析

5.1　传输速率分析

设 V 表示智能合约中无符号整型变量的数量，

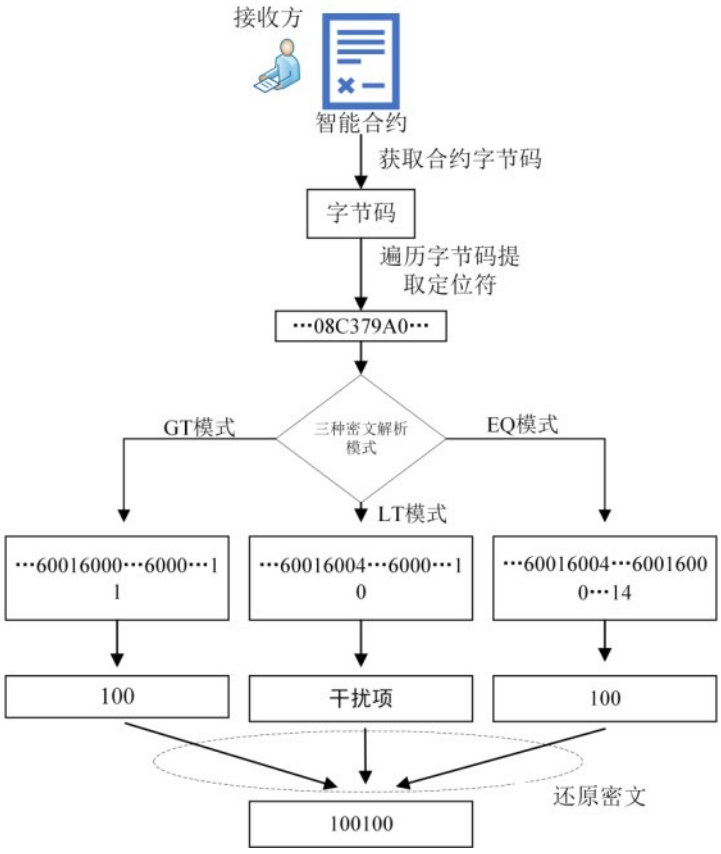


图 5　接收方解码过程

t代表合约部署和交易完成所花费的时间。在不使用LT模式（当前语句充当干扰项）的情况下，通过两两比较可推导出的信息序列最大数量由下式给出：

$$C(V,2)=\frac{V\cdot(V-1)}{2} \tag{1}$$

由于成本和安全性方面的限制，通常难以充分利用所有变量，因此实际生成的信息序列数量范围为[0, C(V, 2)]。

我们从2024年3月至5月期间，从Etherscan已验证合约库 (https://etherscan.io/contractsVerified) 收集了72,526个智能合约。为确保数据质量和一致性，仅保留了源代码已验证且与部署字节码匹配的合约。我们选择了Solidity 0.5.0及以上版本编写的合约，以确保与现代EVM指令的兼容性。此外，排除了代理合约，并基于归一化的字节码哈希值消除了重复项。

所有实验均在一台配备 Intel Core i7-9700 CPU、64 GB RAM并运行Windows10的机器上完成，使用Python 3.11.9。智能合约字节码使用Remix IDE编译，随后使用自定义分析脚本提取操作码序列。

在预处理阶段，我们分析了选定合约中无符号整型变量的分布，以确定嵌入参数V的适当值。总共识别出3,233,007个无符号整型变量。过滤掉在函数作用域内声明的变量后，剩余1,199,832个，平均每个合约有16个可用变量。因此，我们在后续实验中固定V = 16。如第三部分所述，EVM会优先为公共状态变量分配存储空间，其次才是结构体成员。因此，当公共状态变量和结构体变量的数量平衡时，存储布局将达到最优对齐状态，从而在GT和EQ两种嵌入模式下最大化密文容量。

在GT模式下，单个信息序列的信息量范围为[2, 24]比特，其变化趋势如图6所示，平均值通常接近13比特。在EQ模式下，单个信息序列的信息量范围为[2, 6]比特，其变化趋势如图7所示，平均值通常接近4比特。假设系统的传输速率为R，传输时间为t，使用GT模式的次数为i，使用EQ模式的次数为j，且满足约束条件 i + j ≤ C(V, 2)，我们可以计算两种模式获得的平均信息量。系统的传输速率可表示为：

$$R=\frac{13i+4j}{t} \tag{2}$$

在此公式中，13和4分别代表GT模式和EQ模式的平均信息量。这些值基于对大量智能合约数据集的分析，为估算传输速率提供了一个合理的近似值。

5.2　容量稳定性分析

由于发送方可能需要在已有业务逻辑的合约中嵌入信息，实际部署中需要考虑传输容量的稳定性。为此，我们在1000次随机嵌入实验中评估了不同模式下的有效容量分布，结果如表5所示。

表5　　不同模式下的有效容量分布

嵌入模式	平均有效容量(比特)	标准差(比特)	变异系数(CV)
全GT模式	164.2	9.8	6.0%
全EQ模式	51.4	4.2	8.2%
GT与EQ混合模式	142.3	11.7	8.2%
含LT干扰下的混合模式	135.6	12.3	9.1%

实验表明，在本方案中最通用的含LT干扰的混合模式下，平均有效容量达到135.6比特，变异系数为9.1%，低于常见低变异阈值10%。上述结果表示，本方案的隐蔽容量分布集中，具备稳定性。发送方可在部署前，根据合约中可用的空闲存储槽数量S_{free}，及所选模式的平均比特承载量b_{avg}，通过以下模型对可达容量提供保证。

$$C_{\text{guaranteed}}=\min\left(C_{\max},\frac{S_{\text{free}}}{b_{\text{avg}}}\right) \tag{3}$$

其中为C_{max}理论最高容量。该保证机制使得发送方能根据实际需求，在确保隐蔽性的前提下，精确控制并实现目标容量的嵌入。

并且，为验证方案在实际业务场景中的适用性，我们从合约数据集中选取了500个具备完整业务逻辑的已部署合约，在不改变其原有功能语义的前提下，采用含LT干扰的混合模式进行嵌入实验。通过识别并利用未被占用的存储槽或可合并存储的定长变量区域，成功实现了隐蔽信息的嵌入。实验数据显示，94.6%的合约能够成功嵌入不少于100比特的信息，平均隐蔽容量处在(97.3-122.5)比特区间内，源于完整业务合约对存储空间的约束，与实验合约平均容量存在差异，但经Slither安全工具复测，完整业务合约并未引入新的安全问题，证明本

文所提方案具备良好的稳定性。

5.3　成本分析

5.3.1　成本构成与正常交易开销界定

在以太坊平台中，正常交易合约的部署成本主要由交易基础成本、字节码存储成本及构造函数执行成本三部分构成。其中，交易基础成本为恒定数值21000 Gas，用于涵盖将合约创建交易收录至区块链分布式账本的基本计算；字节码存储成本取决于合约编译后生成的字节码规模，依据以太坊黄皮书的定价规则，每存储一个字节的合约代码需消耗200 Gas，正常交易合约的字节码长度通常在2000至5000字节之间，其存储成本相应处于400000至1000000Gas的区间范围内；构造函数执行成本则取决于合约初始化逻辑的算法复杂度与状态变量的写入频次。上述三部分成本的总和共同决定了部署合约所需的最终Gas用量，本文所用方法采用中等代码量级别的交易合约作为信息嵌入载体，包含了正常业务逻辑和变量声明，平均字节码长度为3500字节左右，存储成本处于600000 Gas至800000 Gas区间内，本文取500000 Gas作为基础部署成本。

5.3.2　本方案成本与对比方案成本计算

隐蔽通信方案的经济性是衡量其实际应用可行性的关键指标，涉及从发送方信息嵌入到接收方完成信息提取的全过程开销。

我们将分别计算本方案与对比方案的成本。本文通过在合约中声明变量来嵌入隐蔽信息。我们用 a、b、c 分别表示部署合约所消耗的Gas成本、每增加一个变量所产生的额外Gas成本以及每对变量组合的额外成本。同时，设 i 比特表示每个变量携带的信息量，N 为待传输的密文信息总比特数，p 为实际使用的变量对组合数量与最大可能组合数量的比值（$0 \leqslant p \leqslant 1$）。隐蔽通信模型的系统成本由以下公式给出：

$$Cost_{\text{bytecode}} = a + \frac{N}{i} \cdot b + p \cdot \frac{N(N-i)}{2i^2} \cdot c \quad (4)$$

本文将隐蔽通信成本与文献[11]进行了比较。信息传输成本的构成部分及相应符号如下：相关信息的建立阶段主要用于设置投票合约中的选项。每增加一个选项，Gas成本增加 β。假设每个选项的信息量为 I_O，所需选项数量为 2^{I_O}。因此，为了传输长度为 L_T' 比特的秘密消息，需要 L_T'/I_O 个地址。随后，每个地址的授权成本为 γ，每个地址完成两轮投票或竞价的成本为 θ。三种隐蔽通信方法的成本计算公式如表6所示。

在方法(1)中，设置每个选项的成本 β 随选项数量 2^{I_O} 呈指数增长，过多的选项设置会导致成本急剧上升。在方法(2)中，设置选项没有额外成本，但需要对每个出价进行加密，增加了成本 μ。在方法(3)中，声明每个变量的成本 b 是固定值，而变量组合的额外成本 c 随变量数量线性增加，不会因信

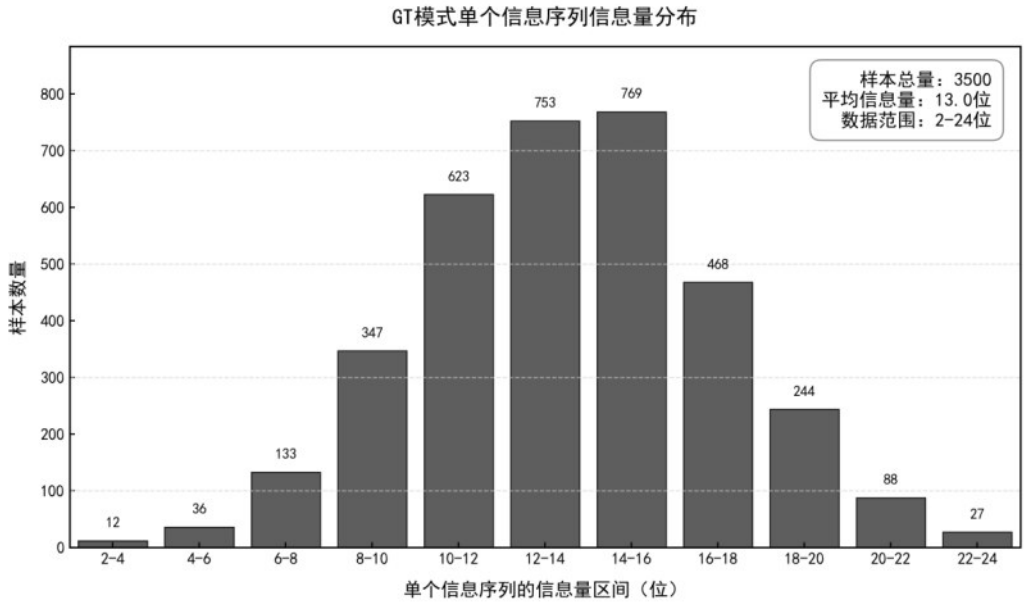


图6　GT 模式下单个信息序列的信息量变化趋势

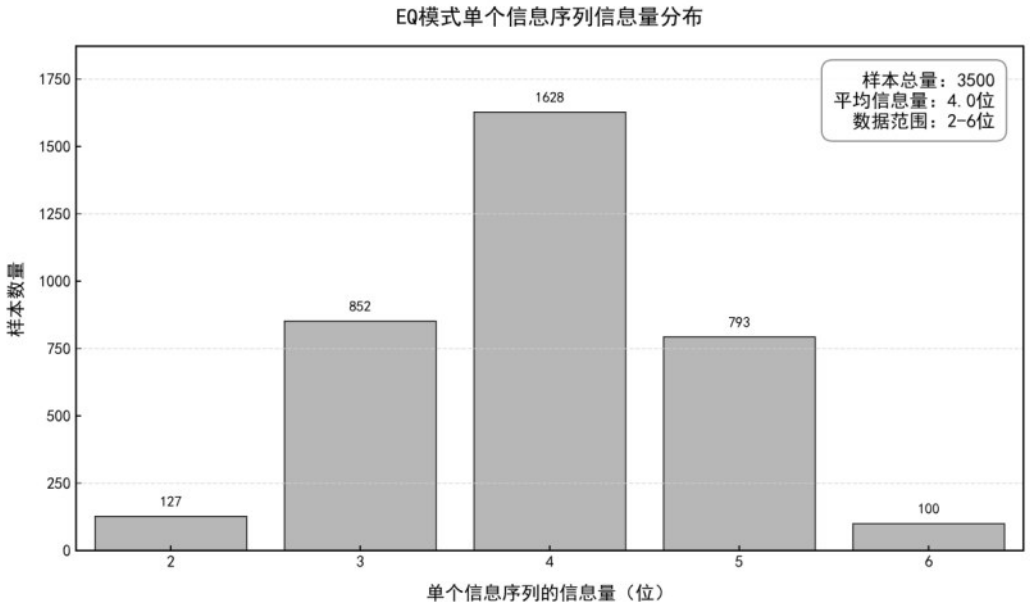


图7　EQ 模式下单个信息序列的信息量变化趋势

表6　三种隐蔽通信方法的成本计算公式

成本类型	公式
$Cost_{vote}$	$\alpha + 2^{I_O} \times \beta + \delta + \frac{L'_T}{I_O} \times (\gamma + \theta)$
$Cost_{\text{hid}}$	$\alpha + \mu + \frac{L'_T}{I_O} \times (\gamma + \theta)$
$Cost_{\text{bytecode}}$	$a + \frac{N}{i} \cdot b + p \cdot \frac{N(N-i)}{2i^2} \cdot c$

息量的增加而发生指数级增长。通过优化变量组合（调整p），可以减少实际使用的变量组合数量，从而降低与变量组合相关的成本，有效减少额外成本c。需要特别说明的是，本方案的成本结构具有非对称性：发送方承担全部嵌入成本，而接收方提取隐蔽信息的过程不消耗任何 Gas 费用。接收方只需根据定位和解码方案从区块链浏览器的公开数据中筛选和提取合约字节码文本，该过程完全在链下离线完成，不产生任何链上交互开销。

为系统评估隐蔽通信方案的经济性，本节选取 128 比特作为标准隐蔽信息负载，在统一的以太坊主网环境参数下（Gas Price =15 Gwei，ETH/USD = 3300），对本方案与两种典型现有方案进行定量测算与对比。

(1)本方案

遵循第 5.2 节给出的理论模型，a 取值涵盖合约创建、基础字节码及初始化逻辑的存储，字节码存储按 200 Gas/字节，综合设定为 500000 Gas。b 表示单变量声明边际成本，根据以太坊黄皮书规则，首次存储一个状态变量开销约为 20000Gas。c 表示变量对处理开销，综合设定 5000Gas。实验采用 GT 与 EQ 混合模式，单变量平均可编码 8 比特信息，为编码 N=128 比特信息，需要变量对 $n = N/i$ = 16，理论最大变量对数为 120，故可得 p 取值 0.133。经公式计算，经济成本为：总 Gas 消耗 899800 Gas，折算美金 44.80USD。根据本文 5.3.1 所述，基础合约部署成本约为 500000Gas，正常交易合约需要声明的状态变量数量大致为 8 到 14 个，声明并显式存储一个变量需要 22000Gas。实验表明，为满足隐蔽通信需求，在利用合约原有状态变量的基础上，需额外声明 4 至 8 个新变量，同时计入 require 比较函数的编写成本。由此，嵌入信息所消耗的额外总成本约为 108250 Gas 至 216500 Gas，折算约为 5.36 USD 至 10.72 USD。若复用合约原有变量进行信息嵌入，可在此基础上降低额外部署成本。

(2)对比方案

优化后的投票合约方案通过合约选项映射信息，其成本随选项数量指数增长。根据原文数据，当采用十六进制信息序列（I_O=4）并使用优化后的多选投票合约时，传输约 10 字节信息的 Gas 消耗约在 8000000 到 9000000 区间。为传输 16 字节信息，我们按线性增长取中值估算，总 Gas 消耗约为 13600000 Gas。

优化后的投标合约方案通过密封投标传递信息，无需设置选项，但需为每轮投标支付加密与交互成本。根据同一文献数据，其成本略低于投票合约。估算传输 16 字节信息的总 Gas 消耗约为 11800000 Gas。

上述三种方案的成本估算结果如表 7 所示。本方案的成本相比两种对比方案降低了约 93%。这主要因为本方案将信息直接编码于一次性部署所生成的字节码中，避免了对比方案中多轮交互、复杂加密验证和大量交易授权操作。此外，本方案成本主体为一次性部署费用，而对比方案的成本来自按地址、轮次和交易不断累加的动态交互费用，会随信息量增长而显著增加。

为了在保持对变量布局精确控制的同时，降低与存储相关的 Gas 消耗，本方案选择性地只初始化那些直接参与密文嵌入的变量。其他已声明的变量保持未初始化状态，并且被排除在合约逻辑和控制流之外。这些未初始化的变量纯粹作为结构占位符，以便根据嵌入过程的要求对齐存储槽。尽管本方案引入了额外的变量声明，但它们可与任何可执行逻辑隔离开来，并且在运行时不被使用。这种设计确保了与现实合约部署的兼容性，并避免了与应用行为的意外交互。我们的实现证实，这种集成不会干扰执行流程，并且与标准的智能合约开发实践完全兼容。此外，如果 Solidity 中未引用的变量未通过合约接口暴露，将不会对执行或可观察到的副作用产生影响。在我们的设计中，所有非功能性变量在外部和内部均不可访问，从而消除了任何相关的安全风险。

根据 EVM 存储的特性，通过使用变量打包技术，可以让多个小尺寸变量共用一个存储槽，这有助于减少存储占用和 Gas 开销。本方案正是利用这种优化紧凑地安排嵌入变量，从而在不增加部署成本的情况下提高嵌入容量。为了验证此布局策略的安全性，我们使用了 Slither 等静态分析工具来检查生成的合约。分析报告未发现与未使用、未初始化或暴露的存储变量相关的问题。这证实了存储布局是安全且确定性的，不会引入攻击面或执行歧义。

5.4 隐蔽性分析

5.4.1 统计隐蔽性分析

Solidity 字节码是由编译器处理的机器码，采用特定编码格式与指令集，对普通开发者而言是不可直接阅读的文本形式。借助这一不可读特性，我们的隐蔽通信过程将密文直接嵌入字节码中，使得检测者难以通过简单阅读字节码破译隐藏信息。本研究通过引入冗余变量，并采用 LT 模式作为干扰因子、使用定位器等策略，有效降低了各变量与生成信息序列之间的关联性，增加了攻击者试图通过数据分析破解隐蔽通信的难度。通过这些方法，我们确保了模型的安全性，使其更能抵御潜在攻击。

为了量化操作码频率分布的稳定性和变化微小性，我们引入以下统计指标：

标准差：衡量数据点相对于平均值的离散程度

$$\sigma = \sqrt{(1/(n-1)\sum\_(i=1)\^n (x\_i - \bar{x})^2)} \quad (5)$$

其中，x_i 为单个数据点，$\bar{x}$ 为样本平均值，n 为样本数量。

变异系数：表示数据相对变异程度

$$CV = \sigma/\bar{x} \times 100\% \quad (6)$$

该指标消除了量纲影响，便于比较不同数据集的稳定性。

变化范围：反映数据的最大波动幅度

$$Range = max(x) - min(x) \quad (7)$$

其中， $\max(x)$和$\min(x)$分别为数据集的最大值和最小值。

相对变化：衡量变化幅度的相对大小

$$Relative\ Change = Range/\bar{x} \times 100\% \quad (8)$$

该指标表示变化范围占平均值的百分比。

上述指标中，变异系数是证明嵌入密文是否影响操作码频率稳定性最直观的指标。根据图 8 至图 11 展示的变异系数值可知，四项高频使用的操作码在嵌入不同数量密文的场景下变异系数的范围是

表 7 传输 128 比特隐蔽信息的成本定量对比

方案	总 Gas 消耗（Gas）	Gas 费用（ETH）	美元成本（USD）	提取成本
字节码方案	约 899800	约 0.0135	约 44.80	链下离线提取，无成本
优化投票合约	约 13600000	约 0.204	约 677.28	链上交互，成本高
优化投标合约	约 11800000	约 0.177	约 587.64	链上交互，成本高

(0.932% - 2.437%)，在统计学中属于低变异的范畴。

字节码的相似性可通过操作码使用频率进行分析。我们准备了多个不同的智能合约，并分析了嵌入0、10、20和30条密文序列样本的操作码使用频率，从中选取了使用最频繁的四种操作码。嵌入不同数量密文序列后各操作码的平均使用频率已在相应图表中展示。Kullback-Leibler（K-L）散度检验用于衡量两个概率分布之间的差异，其定义为：

$$D\_KL(P \mathbin{/\!/} Q) = \Sigma\_x P(x) \log(Q(x)/P(x)) \quad (9)$$

此处，*P(x)*表示未嵌入密文的合约中操作码的概率分布，*Q(x)*表示嵌入了密文的合约中操作码的概率分布。由于操作码数量较多，我们选取了使用频率最高的四种操作码，并对其概率值进行了归一化处理。不同嵌入密文序列数量下的概率分布如图8至图11所示。将未嵌入密文合约与嵌入密文合约的操作码使用频率进行对比的K-L散度计算结果汇总于对应的表8中。

实验结果表明，在嵌入10条密文序列后，合约的操作码使用频率与未嵌入密文的合约高度相似。在嵌入密文序列数为10的条件下，每笔交易最高可隐藏170比特信息。根据算法计算，此时合约操作码使用频率与原始合约的相似度高达99.78%。随着嵌入密文序列数量的增加，相似度略有下降，但整体仍保持在可控范围内。

5.4.2　结构隐蔽性分析

为评估所提嵌入方法的结构隐蔽性，我们对正常智能合约与嵌入了隐蔽信息的合约在操作码层面的3元组模式进行了对比分析。我们收集了200份合约样本，平均分为正常合约组与嵌入密文合约组，并从每组中提取了出现频率最高的前10个3元组操作码序列。

如表9所示，两组间前10个模式中有9个完全相同，重叠率达到90%。常见的模式，例如PUSH2-JUMP-JUMPDEST、JUMP-JUMPDEST-PUSH1和POP-JUMP-JUMPDEST，持续出现在两类合约中，仅频率存在微小差异。

为进一步证实这些结构相似性，图12通过折线图对比了正常智能合约与嵌入式智能合约中最常见3元组操作码序列的归一化频率。

图12的横轴代表按频率降序排列的前几位操作码3元组模式，纵轴则代表其相应的归一化出现率。两条曲线近乎平行的轨迹直观表明了两类合约之间强烈的结构一致性。为量化此相似性，我们采用皮尔逊相关系数这一评估两个分布间线性相关性的标准指标。其定义为：

$$r = \frac{\sum\left(x_i - \bar{x}\right)\left(y_i - \bar{y}\right)}{\sqrt{\sum\left(x_i - \bar{x}\right)^2 \cdot \sum\left(y_i - \bar{y}\right)^2}} \quad (10)$$

其中，x_i 和 y_i 分别表示第i个3元组操作码模式在正常合约组与嵌入式合约组中的归一化频率。基于出现频率最高的前10个操作码模式计算，得到的皮尔逊系数为$r = 0.9997$，对应的p值为 6.42×10^{-14}。这一极高的相关性表明，嵌入过程并未在局部

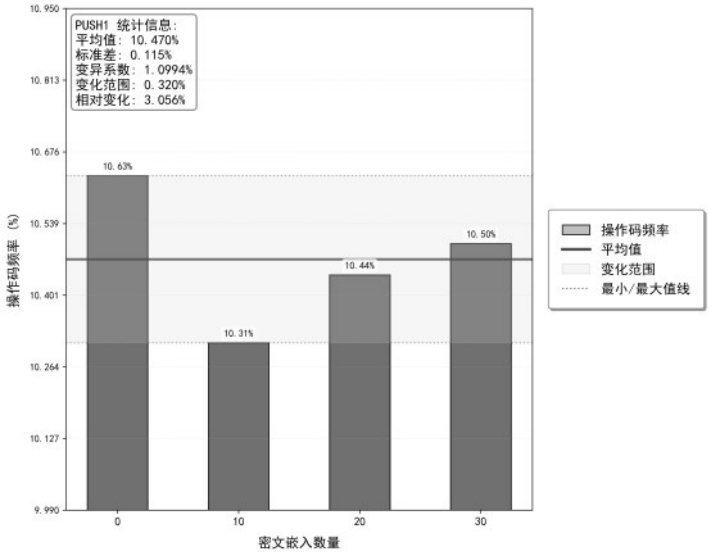


图8　不同密文嵌入数量下的PUSH1操作码频率

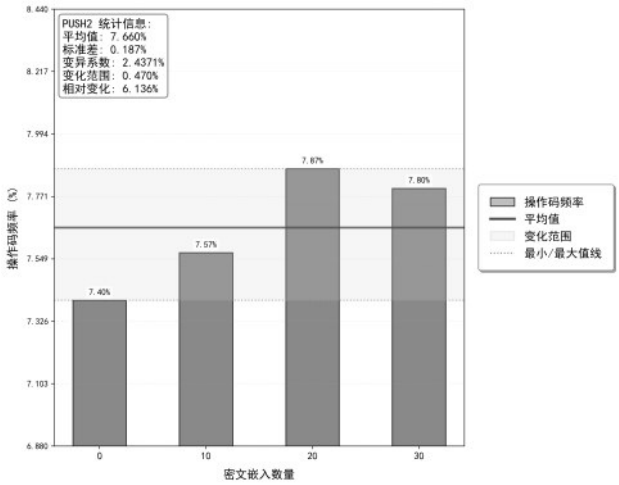


图9　不同密文嵌入数量下的PUSH2操作码频率

指令序列分布中引入统计学上显著的偏差。从结构角度来看，嵌入式合约与正常合约几乎无法区分。这种高度的一致性进一步证实了所提方法在抵抗基于模式的分析和操作码级别检测技术方面的有效性。

5.5　合约代码安全性分析

智能合约代码的安全性指其在设计、编写、部署及执行过程中的安全程度。若智能合约构建过程存在缺陷，可能导致功能不符合预期，造成资金损失或安全风险。本文采用的隐蔽通信方法增加了合约代码量，因此需确保嵌入隐蔽信息后的合约不引发安全问题。

为验证这一点，可采用智能合约安全检测工具。Slither[21]是一款专为Solidity编写的智能合约设计的静态分析框架，能够运行一系列漏洞检测器并提供合约详细信息的可视化报告。Slither可检测编译器可能遗漏的代码优化问题，并提出改进建议。

我们从以太坊官网以及Etherscan随机选取了26个可正常执行的Solidity 0.8.0合约作为样本，使用Slither进行安全检测。这26条合约中有5条未检测出安全问题。其余21条合约中，超五成存在版本约束问题，样本5，样本12，样本13存在缺少零地址验证问题；样本6，样本10，样本11存在状态变化被置为不可声明问题。随后，我们在同一批合约中分别嵌入 5、10 和 15 条密文序列，并再次使用Slither进行扫描。结果显示，嵌入密文前后，所有样本的安全问题类型和数量完全一致，未出现任何新增告警。此外，正常合约与载密合约均能正确执行预定的业务逻辑。这表明，在EVM字节码层面进行信息嵌入不会引入新的安全风险。

此外，我们假设存在一个被动攻击者，其能够检查所有链上数据，包括合约字节码、交易记录和公开可观测参数。该攻击者可能运用静态分析或统计检测技术来识别潜在的隐蔽信道。我们的方法通过维持操作码分布相似性与结构一致性来防范此类检测，这一点已通过KL散度指标和3元组操作码频率对齐性得到验证。

5.6　鲁棒性分析

隐蔽通信的鲁棒性是指系统在面对各类干扰和恶意攻击时，仍能维持隐蔽信息可靠传输的能力。本节从部署前与部署后两个阶段，系统分析本方案

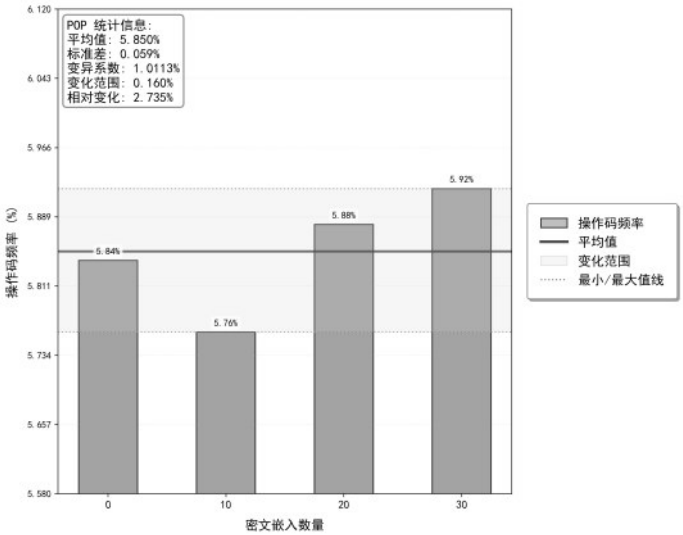


图10　不同密文嵌入数量下的POP操作码频率

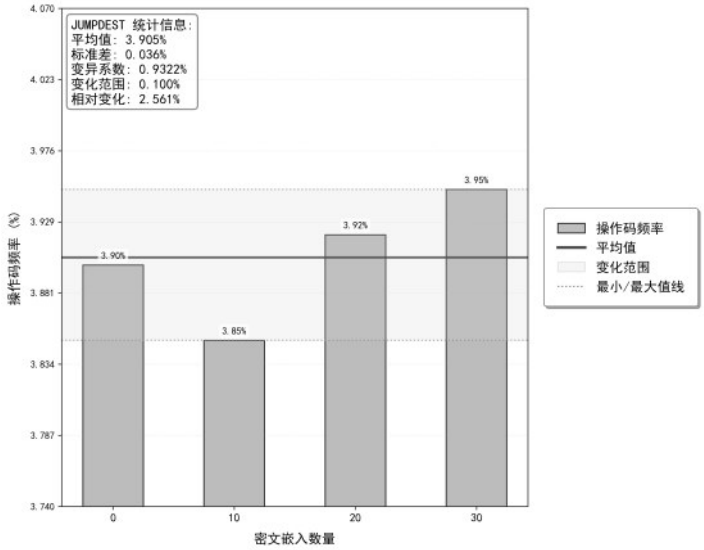


图11　不同密文嵌入数量下的JUMPDEST操作码频率

表8　嵌入不同数量密文序列合约的K-L散度对比

嵌入密文序列的数量	10	20	30
KL散度	0.00080056	0.00223237	0.0121031

表9　正常合约与嵌入合约前10个3-gram操作码序列频率对比

排名	3-gram操作码序列	正常合约频率	嵌入合约频率	频率差值
1	PUSH2-JUMP-JUMPDEST	0.645	0.642	0.003
2	JUMP-JUMPDEST-PUSH1	0.548	0.546	0.004
3	POP-JUMP-JUMPDEST	0.482	0.480	0.002
4	SWAP1-PUSH2-JUMP	0.423	0.421	0.002
5	ISZERO-PUSH2-JUMPI	0.384	0.382	0.002
6	POP-POP-JUMP	0.352	0.351	0.001
7	SWAP2-POP-POP	0.300	0.301	-0.001
8	JUMPDEST-PUSH1-MLOAD	0.252	0.250	0.002
9	MSTORE-PUSH1-PUSH1	0.202	0.201	0.001
10	DUP2-MSTORE-PUSH1	0.152	0.151	0.001

在实际对抗环境下的稳健表现。

在合约部署之前，攻击者可能通过监控合约行为或对字节码进行静态检测来识别潜在的隐蔽信道。本方案通过以下机制增强抗干扰能力：首先，统计相似性维持如5.4节所示，嵌入信息后的合约与正常合约在操作码频率分布上高度相似，Kullback-Leibler散度低至0.0008，高频3-gram操作码序列的皮尔逊相关系数达0.9997，显著降低了基于统计分析的检测成功率。其次，LT模式作为干扰项不携带任何有效信息，但参与存储布局与require语句，增加了攻击者通过模式识别定位嵌入区域的难度。第三，定位器隐蔽性体现在利用require语句错误处理中固定的函数选择器08C379A0作为定位标记，该模式在正常合约的错误处理逻辑中普遍存在，不会因嵌入操作而引入新的可识别特征。

合约成功部署并获得网络最终确认后，其字节码内容永久记录于区块链中，具备不可篡改的特性。这一特性为隐蔽信息提供了以下保障：误码率为零，由于区块链状态一经确认便不可逆转，隐蔽信息在传输过程中不会因信道噪声而发生比特错误，接收方从链上直接获取字节码即可无损恢复原始信息。抗分叉与重组能力方面，尽管以太坊网络可能发生短暂分叉或区块重组，但等待足够数量的区块确认后，交易回滚概率可忽略不计，发送方可选择高确认数策略以保障信息的最终存活性。此外，去中心化存储与冗余确保即使部分节点离线或遭受攻击，接收方仍可通过其他节点同步获取字节码，不存在单点故障风险，显著增强了隐蔽通信的生存能力。

综上所述，本方案在部署前通过统计隐匿性有效抵御检测与干扰，部署后依托区块链的不可变性与最终性实现零误码持久存储，整体系统在对抗环境下表现出强鲁棒性。

5.7　方法兼容性讨论

本文实验主要依赖Solidity0.8.0版本进行，主要基于其代表性及机制稳定性的考虑。实际上，本

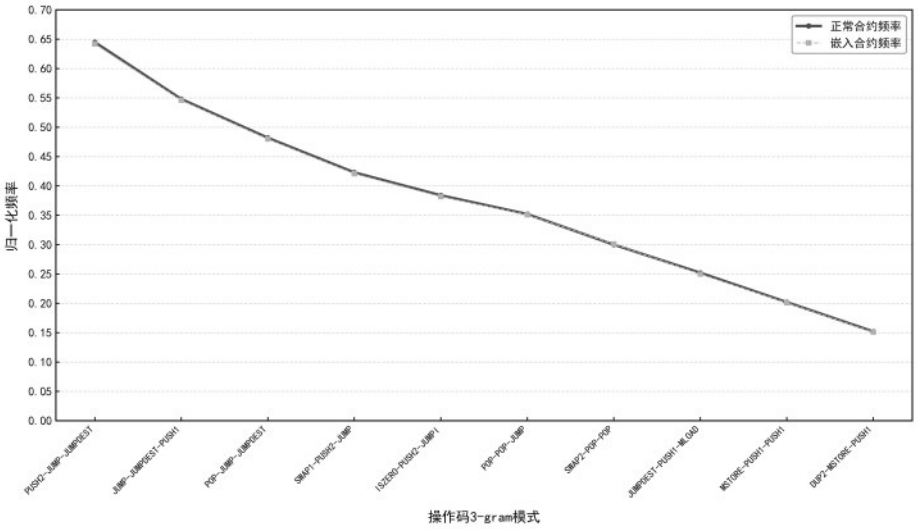


图 12　高频 3-gram 操作码序列在正常合约与嵌入合约中的频率趋势对

方案基于 EVM 字节码的稳定结构特性实现，其核心机制不依赖特定 Solidity 版本，而是作用于编译后的字节码层。

为验证本方法在不同版本中的通用程度，我们在相同实验环境中分别使用 Solidity 0.5.17，Solidity 0.8.0 以及 Solidity 0.8.28 三种版本对同一组嵌入密文的合约源码进行编译和测试。实验结果表明，在所有测试版本中，require 语句均在编译后的字节码中生成了预期的错误选择器 08C379A0。对不同版本生成的“正常合约”与“嵌入合约”字节码进行分析，其高频操作码的频率分布、以及前 10 个 3-gram 操作码序列的皮尔逊相关系数均保持在与 0.8.0 版本实验相近的高水平，这证明嵌入操作并未引入版本特异性的统计异常。不同版本合约对比结果如表 10 所示。

6　结论

本文提出了一种在以太坊字节码中嵌入隐蔽信息的新方案，该方案利用了 PUSH1 指令的高使用频率及其后接单字节操作数的特性。通过运用以太坊虚拟机（EVM）的存储机制，并借助参数的存储分配精确控制 PUSH1 指令的使用次数及其后操作数的数值，我们将密文直接嵌入字节码中。同时，利用 require 函数限定携带隐蔽信息的 PUSH1 指令出现范围，在提升隐蔽通信容量的同时，便于对隐藏信息进行快速筛选。

我们设计了三种密文解析模式及定位器，以适应不同密文量的多种传输需求，使接收方能够从人类无法直接阅读的字节码中提取隐蔽信息。理论分析与实验结果表明，本隐蔽通信方案相比传统方法具有更高的抗篡改性，并兼具优良的传输速率与隐蔽性。

未来工作将致力于解决成本消耗问题，因为过度占用存储空间会产生高昂的 Gas 费用，制约更大量密文的传输。此外，借助 Solidity 使用哈希函数计算动态存储位置的特点，我们可以基于动态值的存储位置嵌入密文，从而进一步提升系统的隐蔽性。

附录本文图 4、图 5 所示的隐蔽通信过程，可通过发布在 Sepolia 测试网上的合约进行验证。登录 EtherScan 并查询交易哈希：（0xa0ea927682321c1cca94abab9d87ad40fe2a54671f53e2343f9471d1fc2971f8），即可获取本文所使用的隐蔽通信方法的字节码。

表 10　　**不同版本合约嵌入结果对比**

Solidity 版本	正常合约 PUSH1 使用率	嵌入合约 PUSH1 使用率均值	是否生成错误选择器	前 10 个 3-gram 操作码序列频率差值范围	Slither 安全检测
0.8.0	10.51%	10.47%	是	[-0.01-0.04]	未引入统计异常
0.5.17	10.58%	10.53%	是	[-0.01-0.03]	未引入统计异常
0.8.28	10.47%	10.45%	是	[0.00-0.03]	未引入统计异常

参考文献：

[1] Rosenthal R. Covert communication in the psychological experiment[J]. Psychological Bulletin, 1967, 67(5): 356.

[2] Deibert R, Rohozinski R, Manchanda A, et al. Tracking ghostnet: Investigating a cyber espionage network[J]. Tracking GhostNet: Investigating a cyber espionage network, 2009.

[3] Chen Z, Zhu L, Jiang P, et al. Blockchain meets covert communication: A survey[J]. IEEE Communications Surveys & Tutorials, 2022, 24(4): 2163-2192.

[4] 张璇, 李雷孝, 杜金泽, 等. 区块链环境下隐蔽信道研究综述[J]. Journal of Frontiers of Computer Science & Technology, 2024, 18(6).

[5] Buterin V. Ethereum white paper[J]. GitHub repository, 2013, 1(22-23): 5-7.

[6] Zou W, Lo D, Kochhar P S, et al. Smart contract development: Challenges and opportunities[J]. IEEE transactions on software engineering, 2019, 47(10): 2084-2106.

[7] Partala J. Provably secure covert communication on blockchain[J]. Cryptography, 2018, 2(3): 18.

[8] 黄冬艳, 李琨. 多地址的时间型区块链隐蔽通信方法研究[J]. 通信学报, 2023,44(2):148-159.

[9] 张宏, 郭云伟. 基于隐蔽通信的访问控制增强技术综述[J]. Cyber Security & Data Governance, 2023, 42(5).

[10] Zhang L, Zhang Z, Wang W, et al. A Covert Communication Method Using Special Bitcoin Addresses Generated by Vanitygen[J]. Computers, Materials & Continua, 2020, 65(1).

[11] Zhang L, Zhang Z, Wang W, et al. Research on a covert communication model realized by using smart contracts in blockchain environment [J]. IEEE Systems Journal, 2021, 16(2): 2822-2833.

[12] 刘九良. 基于区块链技术的信息隐藏方法研究[D]. 南京信息工程大学, 2020.

[13] Liu S, Fang Z, Gao F, et al. Whispers on ethereum: Blockchain-based covert data embedding schemes[C]//Proceedings of the 2nd ACM International Symposium on Blockchain and Secure Critical Infrastructure. 2020: 171-179.

[14] Zhang L, Zhang Z, Jin Z, et al. An approach of covert communication based on the Ethereum whisper protocol in blockchain[J]. International Journal of Intelligent Systems, 2021, 36(2): 962-996.

[15] Basuki A I, Rosiyadi D. Joint transaction-image steganography for high capacity covert communication[C]//2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA). IEEE, 2019: 41-46.

[16] Gimenez-Aguilar M, De Fuentes J M, González-Manzano L, et al. Zephyrus: An information hiding mechanism leveraging Ethereum data fields[J]. IEEE Access, 2021, 9: 118553-118570.

[17] 佘维,程孔,张淑慧,等. 智能合约辅助下的隐蔽通信模型[J/OL]. 计算机应用, 1-12[2026-01-31]. https://link. cnki. net/urlid/51.1307. TP.20250725.1050.002.

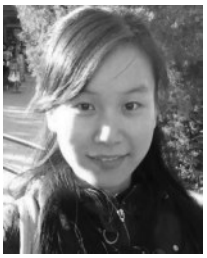
[18] 佘维,马天祥,冯海格,等. 基于合约调用掩盖的区块链隐蔽通信方法[J]. 计算机应用,2025,45(09):2865-2872.

[19] 周志立,丁淳,李进,等. 生成式隐写研究[J]. 计算机学报,2023,46(09): 1855-1887.

[20] 周志立,王美民,杨高波,等. 基于轮廓自动生成的构造式图像隐写方法[J]. 通信学报,2021,42(09):144-154.

[21] Feist J, Grieco G, Groce A. Slither: a static analysis framework for smart contracts[C]//2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB). IEEE, 2019: 8-15.

[作者简介]



黄冬艳 (1984–),女，广西南宁人，桂林电子科技大学副教授，硕士生导师，主要研究方向为区块链性能分析、隐蔽通信、区块链共识算法及物联网。

黄珉 (2001–)，男，广西南宁人，桂林电子科技大学硕士生，主要研究方向为区块链隐蔽通信