Computer Science and Application 计算机科学与应用, 2020, 10(5), 868-882
Published Online May 2020 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2020.105090

Hans汉斯

Research on Trusted Scheme of Sealed Blockchain Auction Based on Ethereum Smart Contract

Jie Yang[1], Yu Sun[2], Feng Liu[2,3*]

[1]Southeast University Chengxian College, Nanjing Jiangsu
[2]Wuxi Taihu University, Wuxi Jiangsu
[3]Shanghai University of International Business and Economics, Institute of Artificial Intelligence and Change Management, Shanghai
Email: *lsttoy@163.com

Received: Apr. 20th, 2020; accepted: May 5th, 2020; published: May 12th, 2020

Abstract

In order to solve the problems of traditional auction methods such as low transparency and high costs and enhance the transparency and credibility of auctions, this paper proposes a sealed auction scheme based on blockchain. Firstly, the goods shall be put on blockchain for bidding and price uncovering; then in order to ensure the security of the transaction funds, the funds trusteeship after the auction is designed in this scheme. According to the logic of multi-signature, the final fund flow will be decided by voting. Meanwhile, this scheme optimizes the auction process by using Keccak256 encryption algorithm, elliptic curve signature algorithm ECDSA and RSA digital signature algorithm by stages. Finally, through the security analysis of the credibility of this scheme, it is proved that the scheme is secure under the condition that smart contracts are not forged.

Keywords

Blockchain, Sealed Auction, Digital Signature, Funds Trusteeship, Smart Contract

基于以太坊智能合约的密封式区块链拍卖可信方案的研究

杨　杰[1]，孙　钰[2]，刘　峰[2,3*]

[1]东南大学成贤学院，江苏　南京
[2]无锡太湖学院，江苏　无锡

*通讯作者。

文章引用：杨杰，孙钰，刘峰. 基于以太坊智能合约的密封式区块链拍卖可信方案的研究[J]. 计算机科学与应用, 2020, 10(5): 868-882. DOI: 10.12677/csa.2020.105090

[3]上海对外经贸大学人工智能与变革管理研究院，上海
Email: *lsttoy@163.com

收稿日期：2020年4月20日；录用日期：2020年5月5日；发布日期：2020年5月12日

摘　要

为解决传统拍卖方式透明度低，成本高等问题，加强拍卖的透明度和可信度，本文提出了一种基于区块链的密封式拍卖方案。首先将拍卖商品放到区块链上进行竞价、揭价，然后为保障交易资金的安全，本方案在拍卖结束后还设计了资金托管，依照多重签名的逻辑投票决定资金最终流向。同时，本方案分阶段使用了Keccak256加密算法、椭圆曲线签名算法ECDSA和RSA数字签名算法来优化拍卖流程。最后对该方案的可信度做了一个安全性分析，证明了该方案在智能合约不被伪造的情况下是安全的。

关键词

区块链，密封式拍卖，数字签名，资金托管，智能合约

1. 引言

拍卖也称竞买，是资本主义制度的一种买卖方式，其良好的价格发现机制、有效的资源分配方式在经济发展过程中发挥了重要的作用。常见的拍卖方式有英格兰式拍卖(open ascending auction)、荷兰式拍卖(open descending auction)、密封式拍卖(first-price sealed-bid auction, FPSB)。其中密封式拍卖是指投标者看不到其他投标者的投标情况，以秘密的方式将投标递交给拍卖人，这样就能够很好地保护投标者的隐私[1]。

然而传统的拍卖方式存在着很多缺陷，比如很难避免出价人用多个身份进行出价的行为；难以避免买方串谋，互相揭示价格以降低商品最终定价的行为；卖方可能会雇佣一部分人来抬高价格等。所以一个必然的趋势是需要寻找一种新型的拍卖方式解决这些问题，随着互联网的发展，电子拍卖的方式便成为了一种可替代的方案。目前，国内外已经出现了许许多多的电子拍卖平台，如eBay、京东拍卖等。但这些交易平台大部分都是中心化的，仍然很难做到每笔交易都公开透明，且平台过分集权容易与买卖方产生利益冲突[2] [3]。

相较于传统拍卖和中心化的电子拍卖，本文方案中使用区块链作为承载拍卖的平台。所谓的区块链，简单来说，引用最早2015年《经济学人》的话就是：“它是一台创造信任的机器。”其典型特点如下：(1) 去中心化的信任机制。系统自身确保真实性，无需外在信任背书主体介入；(2) 稳定性、可靠性、持续性，因为它是一个分布式的网络架构，不存在单点故障，所以相比其他技术架构，它整体性能的稳定性、可靠性和持续性更加突出；(3) 安全的加密机制、共识机制，通过技术中介来达成整个交易，无需第三方的介入；(4) 不可修改特性，高度确保了交易全过程的公开透明和不可篡改性[4]。所以，出于区块链的诸多有利特点，区块链技术是非常适合应用于类似拍卖的多个应用场景。

2. 基于区块链的密封拍卖的方案设计

2.1. 方案介绍以及系统平台的选择

本文中的密封式拍卖方案采用的是维克里拍卖(vickrey auction) [5]。在这种拍卖中，投标者以密封的形式独立出价，商品出售给出价最高的投标者，但是获胜者支付的是所有投标价格中的第二高价，所以也称为第二价格密封拍卖。

为了实践提出方案的可行性，系统解决方案的部署实施放在以太坊区块链上进行。之所以选择以太坊作为测试平台，是因为以太坊区块中不仅可以记录交易、存储数据，还可以在区块中存储一些可执行的代码——智能合约，任何人都能创建合约编写去中心化应用[6]。另外，比特币因区块大小的限制每秒只能处理 7 笔交易[7]，而以太坊的区块大小是没有限制的，所以相对于比特币，在以太坊中进行交易的速度更快，也更适合作为本文拍卖方案依赖的底层平台。

2.2. 密封出价的解决方案

依照维克里拍卖的原理，商品的出价需要进行保密，本方案中基于区块链的密封出价的解决方案流程如图 1。

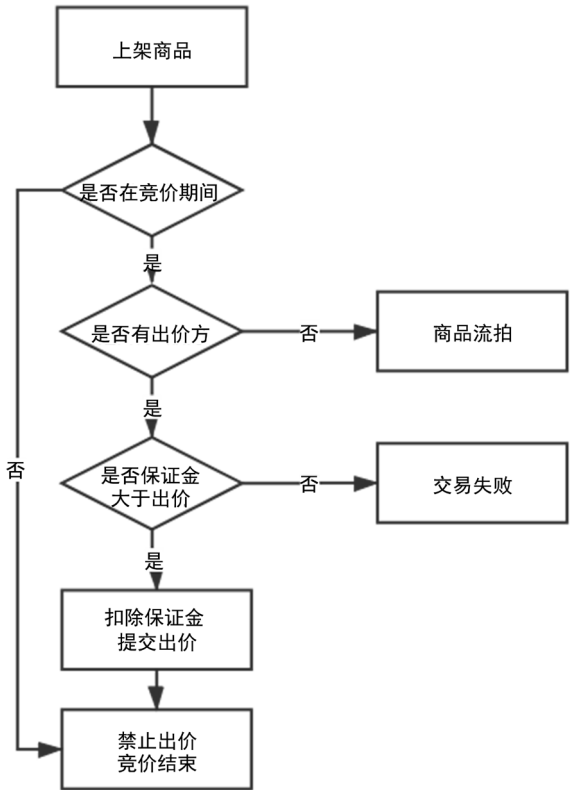

Figure 1. Sealed bid flowchart
图 1. 密封出价流程图

首先，拍卖方先上架一个商品放到区块链上公示，规定商品拍卖的初始金额、拍卖时间以及商品的其他附加信息；然后在拍卖竞价时间内，由不同的出价方对竞价商品进行密封出价，每次出价需要提供己方的真实出价、用来密封出价的密文以及出价需要支付的保证金，此过程中各出价方出价保密且只允许出价一次。如果竞价时间内无人出价，则此次交易流拍；竞价时间截止后，则禁止任何人出价。

2.3. 揭价公示的解决方案

竞价时间结束后，就会进入揭价阶段，此时参与拍卖的各出价方需要输入真实出价和密文，如果报价、报价的密文与之前竞价时一致，退还保证金与真实出价之间的差价；如果不一致，则不会退还；然后揭价截止后，退还其余出价方的出价；最后等待当选仲裁人的第三方宣布拍卖结束，在链上公示胜出的出价方以及交易的金额，揭价公示的主要流程图如图 2。

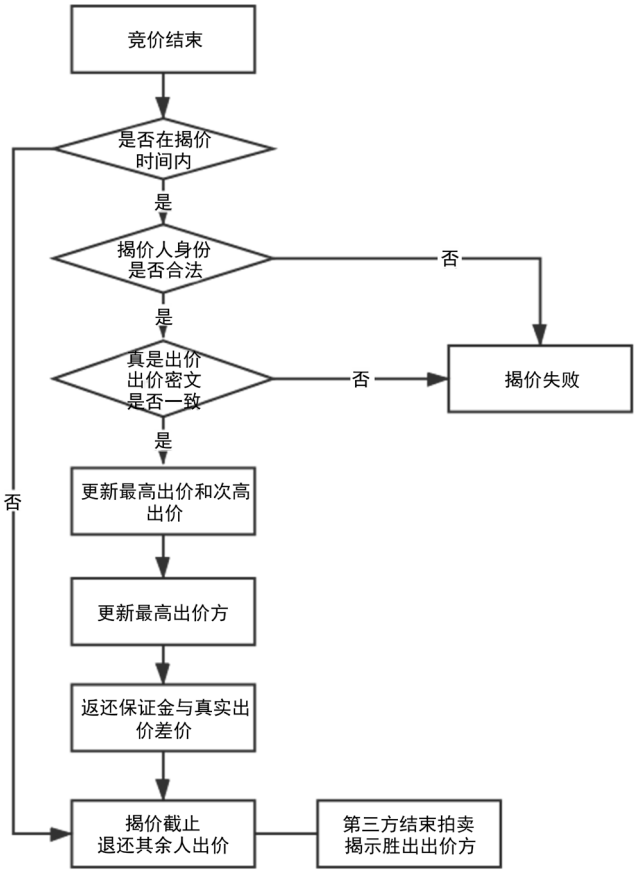

Figure 2. Price uncovering flowchart
图 2. 揭价公示流程图

按照传统密封拍卖的原则，投标是一种以秘密的方式交给拍卖方，然后由拍卖方去执行揭价判断的行为。但如果此过程拍卖方是不诚实的，便会不利于出价方的利益，所以本文方案将揭价的判断交由合约处理，提高一定的交易透明度。当然，在此过程中合约需要保证可信不被篡改。另外，为保证充当仲裁人的第三方的可信度，需要对仲裁人进行一定的资格选定，规定当选仲裁人的第三方需要支付交易金额的 10%作为抵押担保。

2.4. 资金托管的解决方案

实现资金托管是为了规避在去中心化的环境中买卖双方可能遇到的风险，比如卖方不如约交付货物，拿到钱后就跑路；卖方交付货物，但买方不承认收到货物等。同时，把资金分开存放也保障了交易资金的安全。

资金托管的主要流程如图 3。

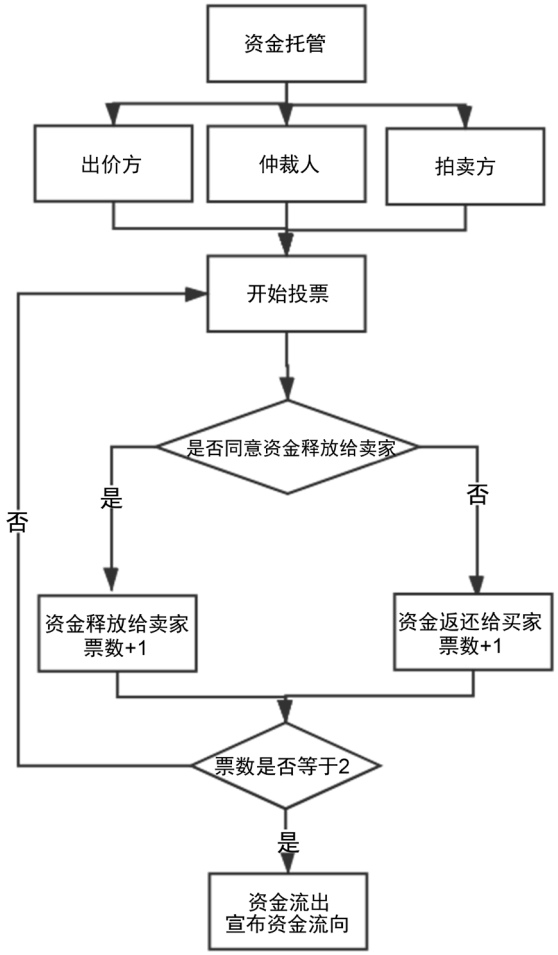

Figure 3. Funds trusteeship flow chart
图 3. 资金托管流程图

依照 2/3 多重签名的设计逻辑，由胜出的买家、卖家、以及仲裁人三方投票决定资金的流向，在三方投票的过程中只要有两方投票同意的资金流向方向一致，资金便会流出。投票结束之后，会在链上公示本次交易资金的流向，同时解除第三方的仲裁人身份。

在资金托管的过程中，让区块链上的其他用户一起监控仲裁人在投票过程中的行为。一旦发现仲裁人有非法行为，就立即中止交易并没收仲裁人的抵押金重新选定仲裁人。如果仲裁人未出现非法行为，则等待资金成功释放给卖家或者返还给买家后，再将抵押金原样返还给仲裁人，同时，支付流转资金的0.1%给仲裁人作为奖励。

3. 方案主要算法的设计及智能合约的交互细节

3.1. 密封出价方案中 keccak 算法的设计

为了保密密封出价中出价方的真实出价，本方案使用了单向密码体制中的 keccak256 算法，只有加密过程，没有解密过程。

keccak256 算法是 SHA3 标准下提出的一种可扩展输出函数 SHAKE128 的核心算法，而完整的 keccak 算法描述比较复杂，由于篇幅有限，本文只介绍 keccak256 加密算法的应用与实现。

keccak 算法[8]采用了海绵结构，基本公式为：

$$b = r + c$$

$$z = SPONGE\left[f\left(b\right), pad, r \right]\left(M, l\right) \tag{1}$$

其中 z 为最终加密输出的密文； $f\left(b\right)$ 为混乱函数，用作置换；b 为计算的内存状态，以矩阵形式存储和运算；r 为会与输入消息和输出摘要发生关系的比特，所有的消息根据 r 长度进行分块，不够分块的就填充 pad；c 为容量，与起始输入消息 M 无关，规定为输出摘要 l 的两倍，l 为消息串 M 的摘要长度。

在本方案中，具体应用的公式如下：

$$M = amount + secretText$$

$$keccak\left[c\right]\left(M, l\right) = SPONGE\left[f\left(b\right), pad\left(10*1\right), r \right]\left(M, l\right)$$

$$sealedBid = keccak\left[c\right]\left(M, l\right).toString\left(hex\right) \tag{2}$$

其中 $amount$ 是出价方输入的真实出价，$secretText$ 是出价方输入的密文，$sealedBid$ 是传输到合约的密封出价，为方便合约存储和判断，需要将加密之后的字符串转化成 16 进制。具体实施步骤为：先在竞价过程中将出价方的真实出价和出价方自己定义的密文进行绑定，然后使用 keccak256 算法对该组合消息进行加密后传给合约进行存储，出价效果图如图 4 所示。

Figure 4. Affect picture of sealed bid
图 4. 密封出价效果图

在揭价阶段再由出价方输入之前出价的金额和密文，交由合约进行比对。合约比对无误后则视为成功揭价一次，揭价的效果图如图 5 所示。

Figure 5. Affect picture of price uncovering
图 5. 揭价效果图

3.2. 交易数据的确认——数字签名

本文为了测试需要，引用数字钱包 MetaMask 作为用户资金存放的工具，图 6 是利用 MetaMask 进行交易的详细数据图。

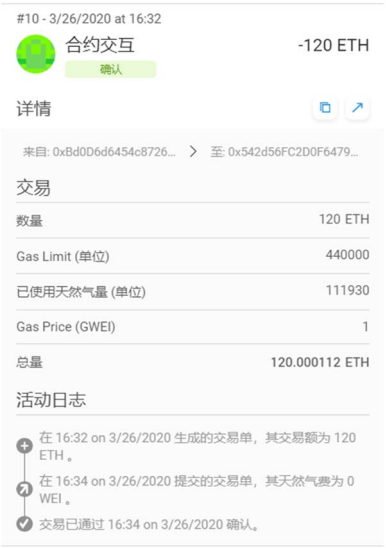

Figure 6. Fund transaction data chart
图 6. 资金交易数据图

虽然在一定程度上使用 MetaMask 钱包保证了账户数据信息的安全，但仍然存在敌手伪造成用户对拍卖交易过程进行破坏的可能性，如敌手可以伪造成出价方身份进行恶意出价，导致出价方无法按理想价格出价。为了降低敌手带来这样类似的风险，在交易逻辑上还需要进一步的改进。

数字签名自 1976 年[9]被提出以来，就作为保障网络信息安全的手段之一，用来解决伪造、抵赖、冒充和篡改问题。本文根据方案设计中的不同情形实现数字签名，对出价方身份及交易数据做进一步确认。

3.2.1. 椭圆曲线数字签名算法 ECDSA 应用与优势分析

椭圆曲线数字签名算法 ECDSA [10]是使用椭圆曲线密码 ECC 对数字签名算法 DSA 的模拟。ECDSA 并不直接对消息进行加解密，而是使用另外安全的散列算法生成数字签名，将公钥经过 hash 运算之后与传入的消息做异或运算。

由于 ECDSA 签名速度快，且密钥存储空间小，在密封出价、揭价这种签名和验签较为频繁的场景下非常适用，另外根据以太坊黄皮书[11]，以太坊已经实现了 ECDSA 算法并进行了封装，配合本文方案适用场景可以直接进行调用。

本文在密封出价和揭价公示时的签名流程和验签流程分别如下：

1) 签名流程：

先选择一条椭圆曲线 $E_p(a,b)$ 和基点 G，对应从中产生的公私钥对为 (e,d)；然后选择临时私有密钥 $k(k<n)$，n 为 G 的阶，利用基点 G 计算点 $(x,y)=kG$；接着产生一个随机整数 $r\equiv x \bmod n(r<n)$，并将消息 M 作为参数，计算 $keccak256(M)$，把得到的比特串化为整数 z 后，计算 $s\equiv k^{-1}(z+rd)\bmod n$，$r$ 和 s 便作为签名值参与验签，如果 r 和 s 其中一个为 0，重新产生r和s。

2) 验签流程：

接收方在收到消息 M' 和签名值 (r,s) 后，计算 $keccak256(M')$，得到的比特串化为整数 z'；然后计

算： $sG+z'e=(x_1,y_1), r_1\equiv x_1 \bmod n$；接着验证等式： $r_1\equiv r \bmod n$，如果等式成立，接受签名，否则签名无效。

因为消息仍然为 keccak256 加密的消息摘要，并不更改之前的交易逻辑，所以此算法的改进加强了出价、揭价过程中的安全。

3.2.2. 资金托管中多重签名逻辑的改进与优化

在资金托管合约中引入多重签名是为了保证资金流转的安全性。因为采用多重签名可以做到防欺诈保护，解决交易双方的信任问题。就相对于“单签名交易”而言，多重签名引入了仲裁人机制，产生纠纷时可由可信第三方做出仲裁，做到交易的不抵赖、付款和出货的保证，从而保障交易的公正性。

多重签名中最常见的就是 2/3 多重签名，然而以太坊外部账户只由一个独立私钥创建，无法进行多签。但是合约具有可编程性，所以可在合约上编写符合多重签名的逻辑代码，实现多个以太坊账户共签交易资金，从而控制资金流向。

如果将用户资金流向选择的消息未经加密而直接传给合约，敌手便可以在传输过程中截取用户的资金流向选择的消息，进而在签名时间内更改用户的选择，最终更改资金流向破坏交易，所以这种方法不可取。因为用户签名之后，合约验签的时候需要对各个用户传过来的资金流向选择的消息加以判断，虽然之前在出价、揭价过程中使用的 ECDSA 签名算法，有签名速度快，密钥存储空间小的特点，但其无法直接对消息加解密[12]，所以本文在这种多重签名的设计逻辑中，选择即可用于签名也可用于加密的 RSA 数字签名算法。

基于 RSA 数字签名体制，使用用户的私钥对资金流向选择的消息进行签名，然后传到合约之后使用用户的公钥进行验签，得到用户对资金流向的判断。RSA 数字签名体制如下：

设 $n=pq$，其中 p、q 为素数，设 $p=q=z_n$，且定义 $K=\{(n,p,q,e,d): ed=1 \bmod \varphi(n)\}$。

对于一个密码体制五元组 $k=(n,p,q,e,d)$，定义：

$$sig_k(x)=x^d \bmod n$$

$$ver_k(x,y)=true\ OR\ false \tag{3}$$

其中 $(x,y\in z_n)$，(n,e) 为公钥，(n,d) 为私钥。因为 RSA 签名的验证过程与 RSA 加解密过程的格式相同，其有效签名验证一致性的成立可以通过 RSA 加解密过程的一致性得到，即：

$$d_k(e_k(y))=x^{ed}=x \bmod n \tag{4}$$

但是如果这样直接照搬 RSA 数字签名的设计，仍然存在伪造他人签名 y，对应的消息是 $e_k(y)=y^e$，一般这种消息是无意义的，但要防止攻击者计算大量的 $e_k(y)$，从而找出有意义的值进行签名伪造。这里给出两种解决的方案，一种是按照密封出价时的逻辑，使用 hash 函数对消息摘要后再签名；另一种是给消息增加可识别的冗余信息后再签名。

对于第一种方案，首先将用户对交易资金流向选择的消息 M 进行 hash 得到消息摘要 $M_1=HASH(M)$，然后使用用户私钥对 M_1 进行签名 $sig_k(M_1)$，再传输至资金托管合约进行验签 $ver_k(sig_k(M_1))$，解密得到 M_1。因为用户资金流向选择只有两种，要么释放资金给卖家，要么返回资金给买家，所以合约验签得到用户资金选择的消息摘要 M_1 后，还要分别对释放资金给卖家的消息 a 和返还资金给买家的消息 b 进行摘要，即

$$M_2=HASH(a) \tag{4}$$

$$M_3=HASH(b) \tag{5}$$

然后将 M_1 与 M_2 和 M_3 进行比对，如果 $M_1==M_2$，那么表示该用户同意释放资金给卖家；如果

$M_1 == M_3$，那么表示该用户同意返还资金给买家。显然，这样资金流转的效率并不高，因为在这种三方参与的多重签名逻辑中，每次资金交易都需要进行至少 6 次 hash 才能决定资金流向(率先签名的两方都同意释放资金或者返还资金的情况)，这样就增加了验签运算过程的时间，从而降低资金流转效率。

对于第二种方案，给消息增加可识别的冗余信息，采用用户对资金流转的消息 M 与当前用户进行签名的时间戳 T 进行标识，然后转成字节码形式后使用用户私钥签名，即：

$$M' = bytes(M + T)$$

$$sig_k(M') = M'^d \bmod n$$

$$ver_k(M') = true\ OR\ false \tag{7}$$

对 M' 进行签名，然后再传给合约使用用户公钥解密后，将拆分开成 M 和 T 两部分，根据 M 判断用户的选择，根据 T 判断用户签名时间的合法性。相比第一种方案，减少了资金流转过程中过多的运算提高了效率，同时依赖时间戳在一定程度上可以防重放攻击，加强了交易的安全性。本文基于第二种方案投票的多签效果如图 7。

Figure 7. Affect picture of voting in funds trusteeship
图 7. 资金托管投票效果图

3.3. 智能合约交互的业务细节

本节将描述方案中与智能合约交互的具体细节，图 8 是方案中的业务交互图。

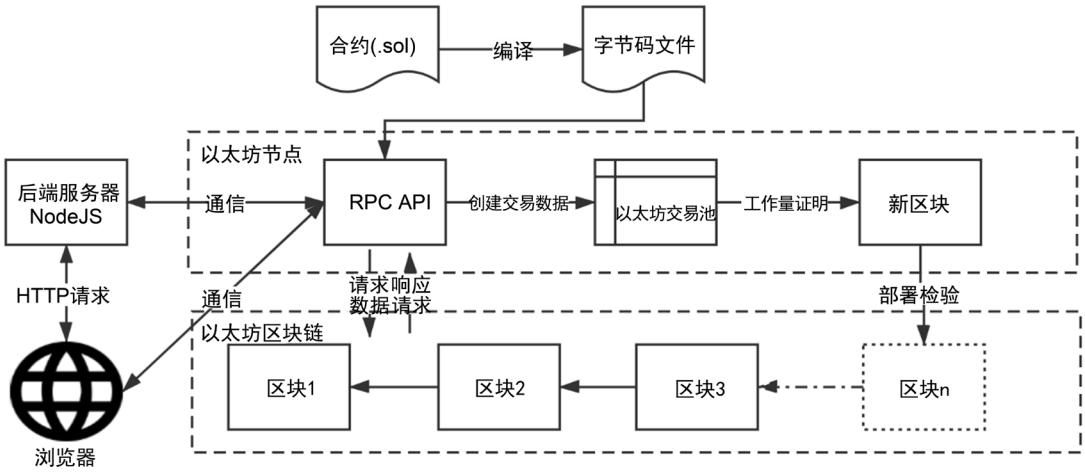

Figure 8. Business interaction diagram of auction scheme
图 8. 拍卖方案业务交互图

如图 8 所示，完整合约交互过程首先应该编写符合规范的合约逻辑，然后将其编译成字节码文件。编译好的字节码文件会提供一些可供远程调用的 API，通过这些 API 创建交易数据暂存到以太坊交易池的入队列中，等待节点上的用户(矿工)利用以太坊共识算法 Ethash 进行工作量证明后，产生一个新区块[13]。最后交由其他节点检验区块的合法性，再部署添加到以太坊区块链上。

方案中通过在以太坊节点上可供远程调用的接口，向区块链请求数据，区块链则会根据不同请求进行响应，接着与浏览器直接进行通信，返回相关数据。当然，也可以通过这些接口先与业务服务器先通信，然后利用 HTTP 请求返回业务数据供前端使用。

3.4. 智能合约交互的技术细节

关于方案中智能合约交互的技术细节，根据 3.3 节描述的业务交互图来看，业务服务器、浏览器在与合约进行交互的过程中，是通过 RPC API 接口读取、写入有关数据的，并没有为了与合约交互而每次都向以太坊网络进行直接请求。当然，编写直接访问以太坊网络的请求逻辑也是可行的，但比较麻烦，管理也很困难。

本文将以拍卖方上架商品与查询拍卖商品信息为例来对合约交互的技术细节进行详细的描述，也就是本方案中通过合约逻辑对区块链数据的读和写的操作。我们通过合约抽象对合约代码进行封装，这样方便在以太坊网络上执行合约函数。图 9 是拍卖方进行商品上架、查询商品数据时的合约的交互逻辑：

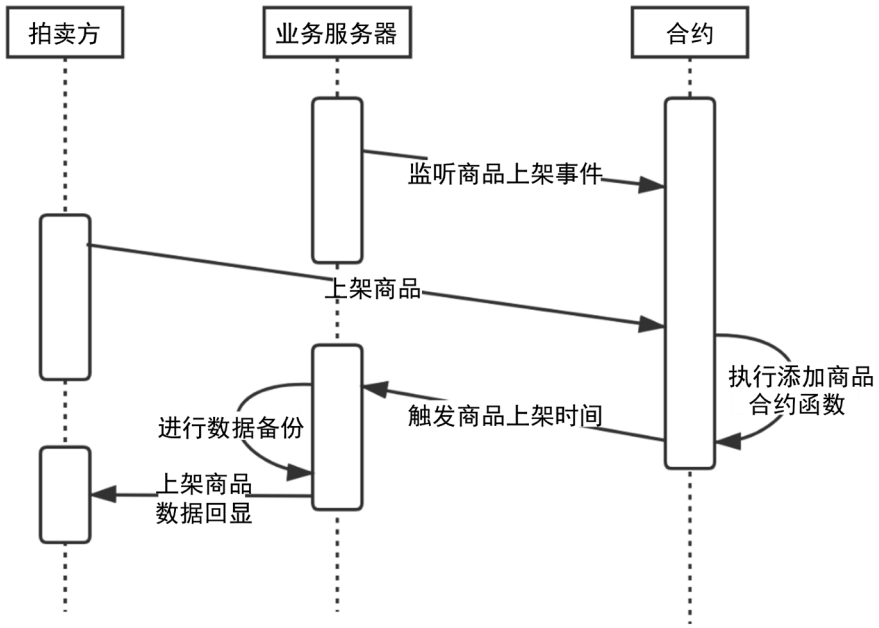

Figure 9. The logic of contract interaction about putting goods for bidding
图 9. 商品上架的合约交互逻辑

图中拍卖方往合约传输商品数据的执行过程就是一个写数据的过程，数据从合约那边返回的过程就是一个读数据的过程。为进一步展示合约被调用的过程，这里给出以太坊私有链客户端 Ganache 上商品上架时生成某个区块里的交易数据内容。

如图 10 所示，在商品上架时调用了 eshop 合约里的 addProductToStore()函数传输商品数据的内容，将数据内容转成 hash 串放入区块，同时触发了 NewProduct 的合约事件，以便交给业务服务器做数据返回的处理。

Figure 10. Details of transaction block
图 10. 交易区块内容详情

4. 安全性分析

因为在解决方案的设计和算法选择上已经根据一些可能存在的攻击方式和安全漏洞加以防范，所以本文第四部分会在第二、三部分的基础上做补充。

4.1. 交易数据的安全性分析

从交易数据的存储上来说，在区块链上传输数据时，相邻的节点会对因交易生成的每个新区块进行工作量证明，验证交易的真实性和合法性。只有验证通过，该区块才能被加入区块链中构成新区块链[14]，确保了数据的可信性。另外，在资金托管过程中，对于每笔交易资金，资金托管的合约地址都是由拍卖合约随机生成的，所以敌手很难确定下一次资金托管合约的地址从而对存储资金合约的地址直接进行攻击。

从数据访问上来说，如揭价公示的商品数据，本文通过 NodeJs Server 调用智能合约中的方法，在 Web 页面上对以太坊区块链上的商品数据进行访问，使得交易信息可信。

从数据加解密的时限上来说，商品的竞价时间、揭价时间都是有一定限制的，竞价结束前不会对出价方身份、出价金额进行验证，揭价结束后也不会再去验证未揭价的出价方身份、出价金额。即规定时间做规定操作，规定时间外的操作无效。交易数据的安全性总结如图 11 的思维导图所示。

4.2. 算法的安全性分析

4.2.1. keccak256 算法的安全性分析

本文中 keccak256 算法用于竞价时出价方出价的密封，此过程中即使敌手在传输过程中截取到密封出价的消息，或者是从合约地址里拿到这段加密消息也无法进行解密。难度规约到对 hash 函数求逆难的

问题，即给定消息摘要 y，找到 x 使得 $h(x)=y$ 困难。另外，由于雪崩效应[15]，对传入的密封出价的消息即使做出一个很小的改变都会产出几乎完全不同的散列值，所以现阶段该算法的应用是安全的。图 12 是利用 keccak256 算法对两个相近的明文值进行摘要的对比图。

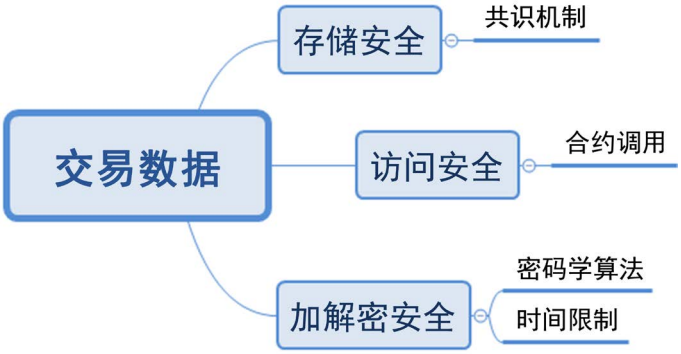

Figure 11. Safety summary of transaction data about sealed auction based on blockchain
图 11. 基于区块链的密封拍卖交易数据的安全性总结

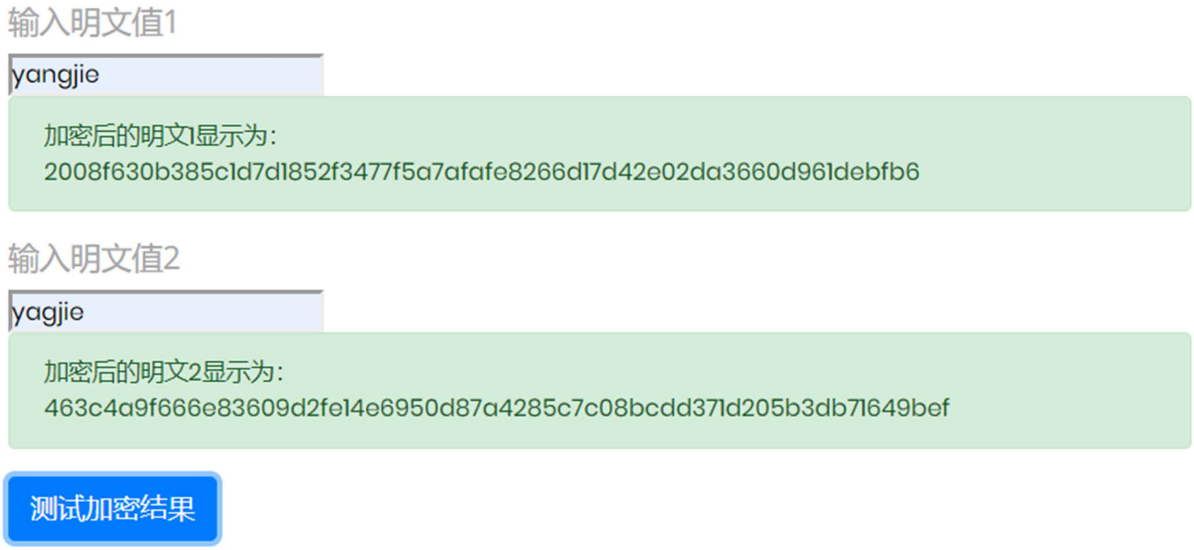

Figure 12. The comparison of keccak256 encryption
图 12. Keccak256 加密对比

4.2.2. RSA 签名算法的安全性分析

因为本文资金托管时使用的 RSA 数字签名体制是基于 RSA 密码体制而设计出来的，所以可以根据针对 RSA 的攻击方式来分析安全性，常见的 RSA 攻击方法主要有分解公开模数 n、共模攻击、小指数攻击、选择密文攻击[16]。

分解公开模数 n 分为直接分解和间接分解。对于直接分解法，根据现有的数据筛法，目前 768 位 RSA 的模数已经被破解[17]；对于间接分解法，可以根据 Pollard 的 $p-1$ 算法[18]来分解 n 获得 n 的因子进行攻击，或者已知解密指数 d 利用随机算法分解 n。对于分解公开模数 nn，最有效的安全方式是牺牲运算效率来提高模数同时注意保密私钥，本文方案中使用 2048 位的模数并利用 MetaMask 保护用户私钥来避免此类攻击方式。

共模攻击则是给定一组群组的用户每个人相同的 n，但指数 d 和 e 不同(公私钥不同)，主要攻击方式为：

1) 群组内的成员，即使不知道n的分解，仍然可以解密其他人的消息。

即给定指数对(公私钥对) (e_1,d_1)，(e_2,d_2)有

$$\begin{cases} e_1d_1 \equiv 1 \bmod \varphi(n) \\ e_2d_2 \equiv 1 \bmod \varphi(n) \end{cases}$$

可以推出：

$$e_1d_1' \equiv 1 \bmod (e_2d_2 - 1) \tag{8}$$

假设可以得到一个d_1'，虽然d_1,d_1'是相同的，但实际上$e_1d_1'-1$也是$\varphi(n)$的整数倍，所以如果别人拿到用e_1所加密的消息，就可以用d_1'来解密这个消息。

2) 群组外人员截获到发送给群组中不同人的同一消息，而这两个加密指数是互素的，则明文可以不用任何一个解密密钥来恢复。

令m为明文消息，两个加密密钥为e_1,e_2，两个密文消息为c_1,c_2，则：

$$\begin{cases} c_1 \equiv m^{e_1} \bmod n \\ c_2 \equiv m^{e_2} \bmod n \end{cases}$$

其中e_1,e_2互素，根据欧几里得算法，$\exists r,s$使得$re_1+se_2=1$，则：

$$c_1^r c_2^s \equiv m^{re_1+se_2} \bmod n \equiv m \bmod n \tag{9}$$

根据公式(9)即可解出明文，所以从安全角度分析，需要避免让一群用户共享模n。

小指数攻击主要分为小加密指数攻击和小解密指数的 wiener 攻击。对于小加密指数攻击，根据文献[16]较小的公钥下发给不同人的同一消息可以利用中国剩余定理解析出明文消息；对于小解密指数的 wiener 攻击，根据文献[19]可以利用辅助参数结合连分数定理计算$\varphi(n)$进行攻击。因此，选择较小的公钥和私钥并不是一个安全的选择。本文方案中 RSA 签名的加密指数为用户的私钥，是借助以太坊私有测试网络 Ganache 随机生成的 256 位长的字节码，用来防止小加密指数的攻击；解密指数为用户的公钥，是借助以太坊私有测试网络 Ganache 随机生成的 160 位长的字节码，用来防止小解密指数的 wiener 攻击。

对于选择密文攻击，本文方案中是对消息加入可识别的时间戳与原消息捆绑在一起，防止敌手虚设部分消息骗取来这些消息的签名，从而对另外的合法消息进行攻击拦截。

表 1 是本方案中对上述攻击的防范方式的总结。

Table 1. The prevention methods of RSA signature algorithm
表 1. RSA 签名算法防范方式

攻击方式	防范方法
分解公开模数 n	选择模数 n 为 2048 位，保密 $\varphi(n)$、n 的素因子和解密指数
共模攻击	每个注册用户使用单独的模数
小指数攻击	加密指数选择 256 位，解密指数选择 160 位
选择密文攻击	交易消息中加入可识别的冗余鉴别码

4.2.3. ECDSA 签名算法的安全性分析

对 ECDSA 签名算法密钥的攻击困难问题可以规约到求解椭圆曲线离散对数问题上，其难度大于大整数分解及有限域上的离散对数问题(ECDLP)。常见的攻击方法有[20]：分解公开模数 n、小指数攻击、MOV 攻击等。

前两种攻击方法与 RSA 签名攻击方法类似，防范方法也不尽相同，这里

就不再赘述。对于 MOV 攻击，主要是针对超奇异椭圆曲线的攻击方法，所以需要在初始签名时选择一条非超奇异的椭圆曲线来防范这种攻击即可。

5. 总结

本文依照维克里拍卖流程加上资金托管，构造出一个基于区块链的密封式拍卖方案，同时在以太坊中测试了该方案具有一定的实用性。本文也对方案的不同流程进行了讨论，选择了合适的加密算法与签名算法，并根据当前存在的典型攻击方式对算法的应用进行了详细的分析，证实了方案在以太坊智能合约不被篡改的情况下是安全可信的。另外，补充介绍了方案中智能合约的交互细节。但由于本人的知识水平和时间精力有限，本方案仍然存在不足，有待后续研究改进，可从以下方面入手：

方案中的数字签名都是单向的，即认定合约地址是可信不可篡改的，尤其是 ECDSA 签名部分，容易遭到中间人攻击，所以如果需要进一步加强安全，可以考虑建立双向的数字签名并申请数字证书，但同时还要注意签名的效率问题；

随着量子计算研究的深入，像 RSA、ECDSA 等算法将会面临被攻破的风险，需要设计更加安全的后量子签名算法来抵抗量子计算的攻击，如全同态加密[21] [22]。

参考文献

[1] Galal, H.S. and Youssef, A.M. (2018) Verifiable Sealed-Bid Auction on the Ethereum Blockchain. In: *Proceedings of International Conference on Financial Cryptography and Data Security*, Springer, Berlin, Heidelberg, 265-278. https://doi.org/10.1007/978-3-662-58820-8_18

[2] 李玮. C2C 电子商务模式的信任修复机制与在线冲突解决的研究[D]: [硕士学位论文]. 北京: 北京交通大学, 2014.

[3] 张帅. 一种密封式电子拍卖方案的构造[D]: [硕士学位论文]. 重庆: 西南大学, 2019.

[4] 刘峰. 区块链热与企业机遇[J]. 企业管理, 2018(6): 19-21.

[5] Vickrey, W. (1961) Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance*, **16**, 8-37.

[6] Ethereum White Paper. A Next-Generation Smart Contract and Decentralized Application Platform. https://github.com/ethereum/wiki/wiki/White-Paper

[7] Kiayias, A. and Panagiotakos, G. (2016) Speed-Security Tradeoff s in Blockchain Protocols.

[8] Bertoni, G., Daemen, J., Peeters, M., *et al*. (2009) Keccak Sponge Function Family Main Document. *Submission to NIST* (*Round* 2), **3**, 320-337.

[9] Diffie, W. and Hellman, M. (1976) New Directions in Cryptography. *IEEE Transactions on Information Theory*, **22**, 644-654. https://doi.org/10.1109/TIT.1976.1055638

[10] Wikipedia (2020) Elliptic Curve Digital Signature Algorithm. https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm

[11] Wood, G. (2014) Ethereum Yellow Paper. https://github.com/ethereum/yellowpaper

[12] 张岩, 张爱丽. 数字签名算法 RSA 与 ECDSA 的比较与分析[J]. 科协论坛(下半月), 2010(2): 102-103.

[13] 松本晋一, 穴田啓晃, 川本純平, 等. 次世代暗号通貨プラットフォーム Ethereum の実験的評価[J]. コンピュータセキュリティシンポジウム 2015 論文集, 2015(3): 1151-1158.

[14] 曹迪迪, 陈伟. 基于智能合约的以太坊可信存证机制[J]. 计算机应用, 2019, 39(4): 1073-1080.

[15] Wikipedia, 雪崩效应. https://zh.wikipedia.org/wiki/雪崩效应, 2019.

[16] 高磊. 基于 RSA 公钥算法的数字签名技术研究[D]: [硕士学位论文]. 绵阳: 西南科技大学, 2015.

[17] Kleinjung, T., Aoki, K., Franke, J., *et al*. (2010) Factorization of a 768-Bit RSA Modulus. In: *Proceedings of Annual Cryptology Conference*, Springer, Berlin, Heidelberg, 333-350. https://doi.org/10.1007/978-3-642-14623-7_18

[18] Pollard, J.M. (1974) Theorems on Factorization and Primality Testing. *Mathematical Proceedings of the Cambridge*

Philosophical Society, **76**, 521-528. https://doi.org/10.1017/S0305004100049252

[19] Wiener, M.J. (1990) Cryptanalysis of Short RSA Secret Exponents. *IEEE Transactions on Information Theory*, **36**, 553-558. https://doi.org/10.1109/18.54902

[20] 王起月. 基于椭圆曲线的数字签名算法研究[D]: [硕士学位论文]. 洛阳: 河南科技大学, 2018.

[21] 何倩. 基于全同态加密的电子投票方案研究[D]: [硕士学位论文]. 杭州: 浙江理工大学, 2018.

[22] 余瑞成. 基于安全比较协议的区块链密封式拍卖方案研究[D]: [硕士学位论文]. 咸阳: 西北农林科技大学, 2019.