



Article

Privacy-Preserving Attestation Scheme for Revocable UAV Charging Using Hybrid State Channels

Xuedan Jia , Xiangmei Song * and Chuntang Yu 

School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China; laura_j@163.com (X.J.); yuct@ujs.edu.cn (C.Y.)

* Correspondence: jlsxm@ujs.edu.cn

Abstract: Although widely applied in varied scenarios, unmanned aerial vehicles (UAVs) suffer severe flight time and flight range limitations due to constrained onboard battery capacity, causing frequent battery recharging when performing persistent missions. The wireless power transfer technology is a promising solution for UAV charging by utilizing unmanned ground vehicles (UGVs) equipped with wireless charging facilities, where charging time slots are auctioned and assigned to UAVs. However, UGVs themselves also have limited energy capacity, resulting in the need to revoke a UAV charging transaction after auction to satisfy their own demand if necessary. In addition, as UAVs and UGVs are mutually distrustful, inherent security and privacy concerns must be resolved during the revocation. In this paper, we resort to blockchain technology for secure and efficient revocable charging in vehicle-assisted wireless UAV networks. We present PAS, an efficient privacy-preserving attestation scheme for revocable UAV charging based on hybrid state channels, where UAVs and UGVs perform off-chain operations as blockchain users for privacy and efficiency, while security and fairness are guaranteed by the on-chain mechanism. PAS consists of a multi-party state channel and multiple two-party state channels responsible for charging scheduling and transaction revocation, respectively. PAS ensures fair and private revocation negotiation and compensation in a trust-free manner by developing a set of carefully designed modular protocols. We provide PAS' constituent primitives in detail, prove its security properties following the universally composable (UC) framework, and present experimental results to demonstrate its feasibility and scalability.

Keywords: unmanned aerial vehicle (UAV); unmanned ground vehicles (UGVs); charging; privacy preservation; revocation; state channel



Citation: Jia, X.; Song, X.; Yu, C. Privacy-Preserving Attestation Scheme for Revocable UAV Charging Using Hybrid State Channels. *Electronics* **2023**, *12*, 3998. <https://doi.org/10.3390/electronics12193998>

Academic Editors: Jingjing Guo, Zhiqian Liu and Linfeng Wei

Received: 9 August 2023

Revised: 18 September 2023

Accepted: 20 September 2023

Published: 22 September 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Featured with low cost, high maneuverability, and rapid deployment, unmanned aerial vehicles (UAVs) can be immediately dispatched to environmentally harsh areas to collect information in an on-demand manner [1]. With these advantages, UAVs have been widely applied in varied scenarios such as disaster recovery and rescue, agricultural production, surveillance, environment monitoring, etc. [2–5]. Due to space and weight constraints, UAVs have inherent battery limitations, restricting their flight time and flight range [6]. Nevertheless, many computing-intensive applications, like video streaming and image processing, require extensive battery endurance, which cannot be addressed by simply optimizing UAV's energy management. Therefore, effective battery recharging is urgently needed for the practical deployment of UAV applications [7–9].

A promising solution to address the UAV charging issue is to utilize wireless power transfer (WPT) technology, including stationary WPT [10], unmanned ground vehicle (UGVs)-assisted WPT [11], and UAV-assisted WPT [12]. Of all these wireless charging solutions, UGV-assisted WPT is a good choice [13], as UGVs are convenient and widespread for on-demand UAV recharging. Shin et al. [11] proposed an auction-based model to allocate UGVs' time slots to UAVs using deep learning. However, UGVs themselves have limited

energy capacity and may need to travel a long distance to perform urgent tasks after the charging scheduling. Despite the easy availability, UGVs face the need to revoke the charging transaction for their own sake, when they have problems providing the scheduled charging time to UAV. In addition, as users in mobile networks are mutually distrustful [14], inherent security and privacy concerns should be fully settled in the open energy trading environment. One main challenge is to conduct autonomously revocable UAV charging with collision avoidance in a secure and private manner.

Blockchain, as an immutable distributed ledger, can facilitate security and transparency in a decentralized and trustless pattern among various entities. It has been adopted in various scenarios, such as charging scheduling [15], data sharing [16,17], and task offloading [18]. The peer-to-peer model of blockchain technology perfectly adapts to UGV-assisted UAV networks. Some recent works have proposed using blockchain for UAVs communication [19,20] and charging [11,21,22]. Although it is proven to be efficient in creating a distributed network for UAV communication and charging, there exist fundamental constraints of blockchain that restrict the use of this technology in revocable UAV charging. First, the inefficiency of on-chain operations is not suitable for energy-constraint UAVs. Second, the transparency property of on-chain information creates privacy concerns. Third, the immutability of the blockchain ledger hinders the fair revocation of charging transactions. Therefore, it is still an open and critical issue to develop a revocable UAV charging scheme with fairness, efficiency, and private on-chain attestation.

In this paper, we aim to realize fair and decentralized UAV charging with reliable revocation, as shown in Figure 1, by resorting to state channel technology. Our framework and method are also suitable for other Internet of Things (IoT) applications with revocability. To achieve our goals, we face two challenges: energy-efficient privacy protection of users' commercial secrets and fair revocation negotiation. The first is how to protect users' commercial secrets from unauthorized exposure during both charging scheduling and revocation procedures with efficiency, and the second is how to realize fair and reliable negotiation and compensation between both trading parties. To address these challenges, we present PAS (a hybrid-channel-based privacy-preserving attestation scheme) for revocable UAV charging, which combines a multi-party state channel and multiple two-party state channels for efficient charging scheduling and fair revocation, respectively. For fairness, we design a specific off-chain revocation protocol. PAS' off-chain protocols and on-chain contracts jointly realize efficient revoking negotiation and reliable revoking attestation in blockchain-based transaction revocation. The main contributions of this work are threefold, as follows.

- A privacy-preserving and fair revocation attestation framework is proposed for the first time based on hybrid state channels. It is suitable for revocable UAV charging. The system framework consists of a multi-party state channel responsible for efficient and private charging scheduling, and multiple two-party channels for reliable and fair transaction revocation;
- To realize private and fair charging scheduling and revocation, we have designed a series of novel and modularized protocols. A two-round revocation negotiation protocol is proposed to balance the efficiency and success ratio of transaction revocation. The smart contract is used for reliable attestation to incentivize and supervise rational and selfish users' legitimate performance;
- We have proved the proposed scheme's security properties under the universally composable (UC) framework, and conducted extensive experiments over a simulated Ethereum network to verify that it can achieve efficient revocation and fair compensation at low system overheads.

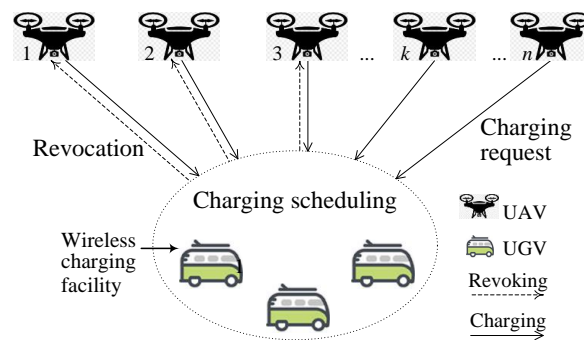


Figure 1. The UGV-assisted UAV charging network with revocability.

2. Related Work

The tamper-proof and immutability properties of blockchain technology create barriers to transaction revocation. Redactable blockchain-based solutions [23–26], aimed at erasing illicit content, can be utilized to delete the recalled transactions. However, these rewriting approaches require massive computation and communication consumption. Because multi-party redaction consensus is necessary to ensure the security of the blockchain system. In addition, they have several shortcomings for transaction revocation, such as the inability to engage in two-party negotiation and fair compensation, the lack of reliable privacy-preserving attestation, and inefficiency. To realize channel recall in the spectrum auction, recall-based schemes have been proposed [27–30], where the sellers preempt the buyers to guarantee their own demand without fair negotiation with the buyers. In [30], compensation is instantly transferred, and the recalled channels can be returned to winners to improve spectrum efficiency and seller utility. However, the recall-based schemes are specific to spectrum sharing and there is no fair negotiation between both trade parties. To guarantee the charging fairness and security between UAVs and UGVs, transactions should be revoked effectively on the UAV’s agreement. Furthermore, privacy-preserving attestation for fair transaction revocation is inevitable. We resort to state channels for secure and efficient resolution.

There are two categories of blockchain channels: a payment channel and a state channel, where the former is limited to payment transactions and the latter is generalized. Many research efforts have been devoted to two-party and multi-party payment channels [31–36], while the state channel has emerged in recent years [37]. The state channel allows users to execute complex smart contracts off-chain efficiently and maintains the blockchain property of trustlessness. Dziembowski et al. [37] first presented the formal definition of the state channel, where only two parties were supported in a channel. It was not applicable to scenarios with multiple parties. Close et al. [38] presented an n -party state channel framework. It could run a restricted set of state channel applications. As a proof of concept, they implemented the protocol for the two-party case on Ethereum. McCorry et al. [39] presented an application-agnostic state channel construction, Kitsune, which supported n parties and allowed the channel to be turned off such that the application’s progress can continue via the blockchain. However, an honest party will not use the dispute process if it is too costly. This will clearly hamper real-world use of state channels [40].

Recently, Nguyen et al. [41] extended the original concept of state channels to support multi-party computation, in order to realize an iterative double auction. They presented a decentralized and trustless auction framework without a trusted third party, where a double auction can run efficiently on a blockchain network without high latency and on-chain fees. The resulting protocol is not revocation compatible. For spectrum recall, Liu et al. [30] proposed a state channel-based method where instant compensation can be transferred efficiently in the channel. The recall-based schemes are specific to spectrum sharing and there is no fair negotiation between both trading parties.

In this work, we establish a secure and fair UAV charging system with revocability based on hybrid state channels. First, we establish a multi-party channel that can support

privacy-preserving trade matching among multiple UGVs. In addition, we also provide additional functionalities to guarantee the creation of two-party state channels between UGVs and UAVs where the private revocation is performed. Based on this hybrid state channel framework, we design our privacy-preserving attestation scheme and prove its security properties in the UC model.

3. System Model and Design Goals

Before working with the specific scheme, we describe in detail the system model, the threat model, and our design goals in this section.

3.1. System Model

We introduce a privacy-preserving transaction revocation model based on hybrid state channels, as illustrated in Figure 2, where multi-party state channel *Mchl* is for charging scheduling and two-party state channel *Tchl* is for transaction revocation. We consider a set of sellers (UGVs) *S* and buyers (UAVs) *B* connected to a blockchain network. We denote $U = B \cup S = \{U_1, U_2, \dots, U_n\}$ as the set of *n* users. Buyers require energy resources from sellers and the trades are matched in *Mchl* among all users. On creating *Mchl*, each user should pay a deposit of amount $(\alpha + \beta)$, as shown in Figure 3— α for the charging scheduling process and β for creating *Tchl*, respectively. The deposit is large enough and used to prevent the users from aborting or behaving maliciously. If losing the auction with no trade matched, a user gets all the deposit $(\alpha + \beta)$ back after the matching stage. For winners in the matching stage, part of the deposit β is used to reinforce the creation of a two-party channel *Tchl* for trading or revocation. We adopt the double auction scheme proposed in [42,43] as the off-chain trade matching protocol. Different from [43], the auction protocol is conducted off-chain by UGVs without interaction with UAVs or the blockchain. Finally, the scheduling result is submitted to the blockchain by the seller before the channel is closed. The other part of our framework is the transaction revocation module, which is used when there is a revocation requirement, i.e., neither charging (①) nor payment (②) is performed. The seller who needs to recall the sold charging time proposes to revoke the transaction matched in the previous procedure. The winning seller and the corresponding buyer construct a two-party state channel and conduct the revocation negotiation and compensation in the channel. During the creation of *Tchl*, each winner should also pay a deposit of amount γ for fair revocation. Note that the deposit α, β, γ is large enough according to the blockchain deposit mechanism, i.e., $\alpha \gg bid, \beta \gg p_{ij}, \gamma \gg c_{ij}$, where *bid*, *p_{ij}* and *c_{ij}* are the user bid, clearing price, and revocation compensation, respectively.

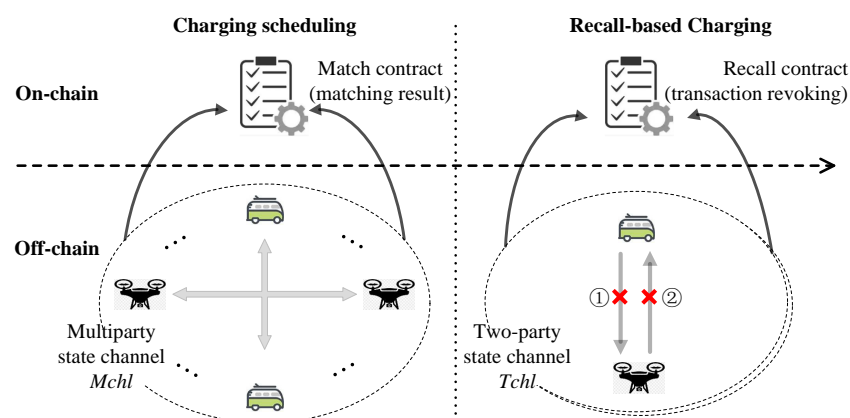


Figure 2. The revocable transaction framework based on hybrid state channels.

The main idea of this paper is to construct an efficient and private trading system with revocability and realize privacy-preserving attestation for secure and reliable UAV charging. Under the proposed framework, we introduce the privacy-preserving attestation

scheme PAS for revocable UAV charging. The specific PAS sequences are as shown in Figure 3, where the precise protocols and functionalities are elaborated in the next section.

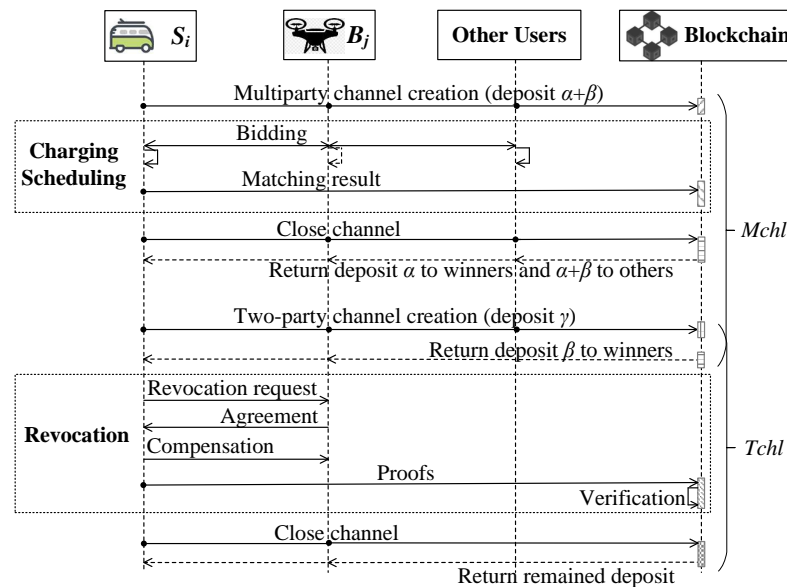


Figure 3. The sequence diagram of PAS process.

3.2. Threat Model and Design Goals

In our content, we consider the issue of fair and privacy-preserving transaction revocation. First, buyer and seller take an equal place in the revocation rather than one being prior to the other. That is, to successfully revoke a matched trading, both the buyer and the seller must reach an agreement on the revocation and have no dispute over the compensation. Otherwise, the charging transaction is forced. Second, from the perspective of blockchain-based application regulation, the revocation can be verified and attested in a privacy-preserving manner to further guarantee revocation fairness, i.e., privacy-preserving attestation on the revocation is necessary.

We consider selfish and rational sellers and buyers, i.e., all users aim at maximizing their own profit and speculating on others' private information. We assume that U is registered to the regulation authority before joining PAS. All users in PAS communicate with each other via a secure and authenticated off-chain channel. Each user and the ideal functionality will automatically discard any messages originating from a user that is not in U or when the message's signature is invalid.

We consider a computationally efficient adversary, \mathcal{A} , who can corrupt any subset of users, acquire their internal state and messages, and send messages on behalf of the corrupted users.

The blockchain is represented as an append-only ledger L that is managed by a global ideal functionality \mathcal{F}_L (same as [37]). Blockchain nodes are honest but curious [16]. They perform according to the blockchain rules and try to infer others' private information. In other words, we believe that the blockchain is secure and on-chain information is publicly available without privacy. For privacy preservation, on-chain operations should reveal no private information.

For simplicity, we assume that all users have enough funds in their accounts to make deposits to the smart contract. Furthermore, we assume a synchronous communication network. A round is a unit of time corresponding to the off-chain communication delay between a pair of users. Any modifications on \mathcal{F}_L and smart contracts take at most $\Delta \in \mathbb{N}$ rounds. Δ means that updates on the blockchain are not instant but can be completed within a predictable amount of time. Moreover, each user can retrieve the current blockchain state of \mathcal{F}_L and smart contracts in one round.

With respect to the threat model, we define the following goals of interest:

(1) Privacy Preservation. First, in the charging scheduling stage, all users' private information, like bid, charging time, charging amount, and charging location, should be hidden from each other and outside adversaries. Second, during the transaction revocation, the negotiation and compensation reveal neither users' identities nor bids in the former stage. Third, the on-chain verification and attestation disclose no user's private information.

(2) Fairness. The buyer and seller take an equal place in the revocation process. The seller cannot refuse to provide the charging service, while the buyer can not refuse to pay without the other's agreement on the revocation. Only when both the buyer and seller reach a consensus on the revocation, is it taken as legal; otherwise, the misbehaved user will lose all the deposit. What is more, participation fairness can also be guaranteed for all users during the off-chain scheduling according to [16], which is not the focus of this paper and we do not extend it.

(3) UC Security. Both on-chain and off-chain operations are necessary under the PAS framework, so the security of the scheme can be guaranteed when the on-chain and off-chain protocols can perform safely under the universally composable (UC) framework.

(4) Efficiency. The overhead of realizing the above goals should be acceptable from the perspective of system users, especially for UAVs. The charging scheduling and revocation processes should minimize communication and computation overhead, instead of repeating the costly auction assignment and revocation negotiation by all users on-chain.

4. Privacy-Preserving Attestation Scheme for Blockchain-Based Transaction Revocation

We first present the ideal functionality of PAS that defines how a revocable trading system is operated using the hybrid state channels. We then describe the designed protocols that realize the ideal functionality.

4.1. Ideal Functionality

The ideal functions include the ledger's ideal functionality \mathcal{F}_L and the ideal PAS functionality \mathcal{F}_{PAS} . First, we define \mathcal{F}_L , which supports adding (returning deposit) and subtracting (removing deposit) one's balance, as shown in Figure 4.

<p>Ideal Functionality \mathcal{F}_L <i>Remove and return deposit</i> On receiving <i>update</i>(U_i, x) (1) If $x > 0$, $Ledge[U_i] = Ledge[U_i] + x$ (2) If $x < 0$ and $Ledge[U_i] \geq -x$, $Ledge[U_i] = Ledge[U_i] + x$ (3) Otherwise, return <i>error</i>() and stop.</p>
--

Figure 4. The ideal ledger functionality \mathcal{F}_L .

As we have mentioned, PAS consists of two stages: the matching stage and the trading/revoking stage, respectively, implemented in channels $Mchl$ and $Tchl$. The formal definition of \mathcal{F}_{PAS} is presented in Figure 5. It supports the following seven functionalities: open multi-party channel, determine trade matching, return deposit, open two-party channel, close multi-party channel, trade revocation, and close two-party channel.

(1) Open multi-party channel. All buyers and sellers joining the same auction process create a multi-party state channel by sending a message *create*() to the judge contract. After all users agree on creating the channel by sending *create*(), the channel is successfully created. The contract removes a deposit ($\alpha + \beta$) from the account of each user.

(2) Determine trade matching. Each user first handles the true bid value using distributed encryption, then broadcasts their bidding information, including the bid, battery amount, charging time, and location. With all the bids collected, charging scheduling can be performed in a privacy-preserving manner by sellers according to the double auction model introduced in [43].

(3) Deposit return. If no user has disputes on the auction result, they proceed to submit the trades matched to the judge contract. On receiving the matching result, function

Return() is performed to return the deposit to the corresponding users' account except that part of the winners' deposit β is reserved to enforce the trading.

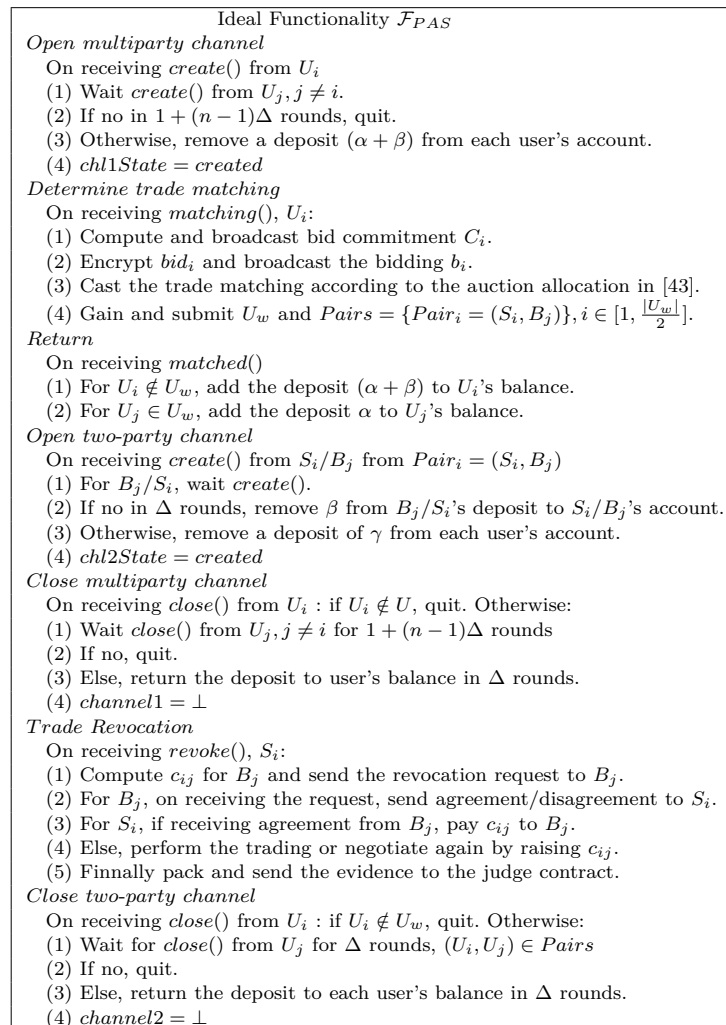


Figure 5. The ideal PAS functionality \mathcal{F}_{PAS} .

(4) Open two-party channel. The winning seller and buyer create a two-party state channel to conduct the trading or revocation, and they deposit γ for fairness and misbehavior punishment. If any winning user fails to behave honestly, i.e., create the trading channel in time, it will lose the deposit β . After the trade is matched, no winning user is allowed to abort the system, or it will lose β . Meanwhile, if the seller has a problem delivering the sold item, i.e., charging the winning UAV, it negotiates with it and pays the compensation in the channel.

(5) Close multi-party channel. After all winner pairs have created the two-party channels, the multi-party channel can be closed on the agreement of all users by sending *close()*. Otherwise, the channel remains open for a period, after which winners with no two-party channel created will lose their deposit. On multi-party channel closure, the winning buyers and sellers can get their deposit back.

(6) Trade revocation. If any sellers cannot deliver the sold item after the matching, due to urgent energy consumption, like instant tasks or long traveling needs, they send a revocation request in channel *Tchl* to the corresponding buyer. The buyer can agree with or reject the request for its own sake. Fair and private negotiation and compensation are conducted in *Tchl*. Note that when there is no need for transaction revocation, the seller and buyer execute the trading. After the revocation or trading, the corresponding proofs are submitted to the judge contract, which then performs the verification.

(7) Close two-party channel. If the trading or revocation has been successfully conducted, the channel comes to its termination procedure upon receiving *close()* from both users. On closing *Tchl*, the remained deposit of each user is returned to the corresponding account.

4.2. Protocol for PAS

In this subsection, we present the detailed protocol design based on hybrid channels that realize fair and reliable charging with revocability. The PAS protocol consists of two main parts: the off-chain protocol and the on-chain judge contract. The key function of the judge contract is to regulate the state channels and perform verification of the trading or revocation. The off-chain protocol is operated among users joining a specific state channel. In PAS, we have multiple channels of two different kinds: one multi-party channel and a couple of two-party channels. All users perform *create()* and *close()*, respectively, to establish and withdraw from the corresponding channel. Apart from the creation and closure of the channel, users perform two off-chain protocols: charging scheduling and trade revocation. All users conduct the matching protocol in the multi-party channel to reach consensus on the matching result. The winning user pair creates a two-party channel to execute the trade or perform negotiation to revoke the transaction.

4.2.1. Judge Contract

The detailed judge contract is shown in Figure 6. One of the key features of \mathcal{F}_{Judge} is to regulate the multi-party channel and the two-party channels. Note that, after the matching result is submitted, channel *Mchl* cannot be closed at once. To guarantee fair trading and revocation, the two-party trading channel between the winning seller and buyer must be created. If all two-party channels are created, the multi-party channel can be successfully closed on all users' agreements. The precise creation and closure of the channels can refer to [37,41]. The other functionality of \mathcal{F}_{Judge} is to verify and regulate the trading/revocation processes. The precise verification function is illustrated in Algorithm 1. As on-chain records are immutable, users cannot deny the trading result. The judge contract guarantees the properties of public verifiability and fairness.

Judge Contract \mathcal{F}_{Judge}	
<i>Open multiparty channel</i>	
On receiving <i>create()</i> from U_i , wait <i>create()</i> from $U_j, j \neq i$.	
(1) If not receiving after $1 + (n - 1)\Delta$ rounds, stop.	
(2) Else, set the channel parameters and remove deposit $(\alpha + \beta)$.	
<i>Matching result submission</i>	
On receiving <i>matched()</i> from U_i , return the deposit:	
(1) For $U_i \notin U_w$, add the deposit $(\alpha + \beta)$ to U_i 's balance.	
(2) For $U_j \in U_w$, add the deposit α to U_j 's balance.	
<i>Open two-party channel</i>	
On receiving <i>create()</i> from S_i from $Pair_i = (S_i, B_j)$	
(1) For the other user B_j , wait <i>create()</i> .	
(2) If no in Δ rounds, remove β from B_j 's deposit to S_i 's account.	
(3) Else, move deposit γ and set <i>flag</i> = <i>NotAttested</i> .	
<i>Close multiparty channel</i>	
On receiving <i>close()</i> from U_i : if $U_i \notin U$, quit. Otherwise:	
(1) Wait <i>close()</i> from $U_j, j \neq i$ for $1 + (n - 1)\Delta$ rounds	
(2) If no, quit.	
(3) Else, within Δ rounds, return deposit and set <i>channel1</i> = \perp .	
<i>Trading result submission</i>	
On receiving <i>trading()</i> from U_i	
(1) Perform the verification as Algorithm 1 and publish the result.	
<i>Close two-party channel</i>	
On receiving <i>close()</i> from U_i : if $U_i \notin U_w$, quit. Otherwise:	
(1) If <i>flag</i> = <i>NotAttested</i> , quit.	
(2) Within Δ rounds, wait for <i>close()</i> from $U_j, (U_i, U_j) \in Pairs$	
(3) If no, quit.	
(4) Else, within Δ rounds, return deposit and set <i>channel2</i> = \perp .	

Figure 6. The judge contract \mathcal{F}_{Judge} .

Algorithm 1 Revocation Verification**Input:** Seller–buyer pair (S_i, B_j) , evidence for the trading result $trading = (Type, Evidence)$ **Output:** Verification result $Attested$

```

1: if  $(S_i, B_j) \notin Pairs$  then
2:   Quit
3: else
4:   Verify  $S_i/B_j$ 's signature  $\sigma_i/\sigma_j$  in  $Evidence$ .
5:   if  $Ver(\sigma_i) = \perp$  then
6:     Confiscate  $S_i$ 's deposit  $\gamma$ ;  $Attested=S_i$  cheated.
7:   if  $Ver(\sigma_j) = \perp$  then
8:     Confiscate  $B_j$ 's deposit  $\gamma$ ;  $Attested=B_j$  cheated.
9:   if  $Type = Rev$  then
10:    Verify  $S_i$ 's transfer of  $c_{ij}$ .
11:    if  $Transfer(c_{ij}) = \perp$  then
12:      Confiscate  $S_i$ 's deposit  $\gamma$ ;  $Ledger[S_i] = Ledger[S_i] - c_{ij}$ ;
13:       $Ledger[B_j] = Ledger[B_j] + c_{ij}$ ;  $Attested=S_i$  paid no compensation.
14:    else
15:       $Attested=Revocation$  completed.
16:     $flag = RevAttested$ 
17:  else  $Type = Dev$ 
18:    Verify  $S_i$ 's delivery and  $B_j$ 's payment.
19:    if  $Deliver_{ij} = \perp$  then
20:      Confiscate  $S_i$ 's deposit  $\gamma$ ;  $Attested=S_i$  did not deliver the item.
21:    if  $Payment_{ji} = \perp$  then
22:      Confiscate  $B_j$ 's deposit  $\gamma$ ;  $Attested=B_j$  did not pay.
23:    if  $Deliver_{ij} = \top$  &  $Payment_{ji} = \top$  then
24:       $Ledger[S_i] = Ledger[S_i] - p_{ij}$ ;  $Ledger[B_j] = Ledger[B_j] + p_{ij}$ ;
25:       $Attested=Delivery$  completed.
26:     $flag = DevAttested$ 

```

4.2.2. Off-Chain Protocol

Here, we provide the details of the off-chain charging scheduling protocol and the transaction revocation protocol. We eliminate the detailed channel creation and closure processes for the sake of space here. We refer the readers to [37,41] for the precise protocols for the creation and closure of the multi-party channel and two-party channel. The matching protocol is conducted among all users in the multi-party channel to get the matching result, where all winning sellers have enough energy to meet the charging requirements of buyers; otherwise the sellers will lose in the charging scheduling. Then the winning users create $Tchl$ in pairs to perform the trade or negotiate to revoke the transaction.

(i) Charging Scheduling Protocol: The charging scheduling protocol in PAS contains commitment [44], bidding, allocation [45], and result submission. The precise steps are depicted in Figure 7, where the bidding information b_i includes the bid, battery amount, charging time, and location. As the focus of this paper is on revocation and there are many works on charging scheduling, we ignore the details of the matching process. The detailed privacy-preserving double auction process can be found in [43]. During each step, if some user does not follow the protocol, the eccentric state will be submitted to the judge contract through $submit()$. Then the process will be stopped, and the malicious user will be punished by losing the deposit. If the matching result is submitted, it means that all users agree on the trades and transactions. From now on, no winning user is allowed to withdraw before accomplishing the trading or revocation; otherwise, they will lose all the deposit as a punishment.

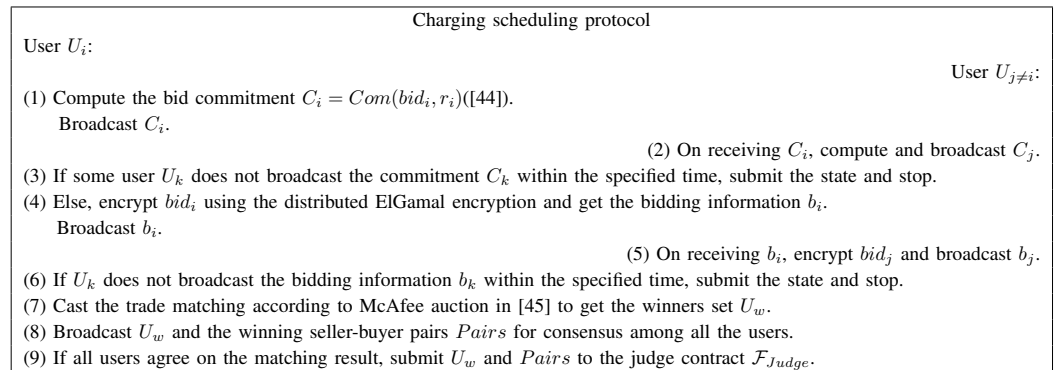


Figure 7. The charging scheduling protocol.

(ii) Transaction Revocation Protocol: The transaction revocation protocol consists of revocation request, revocation response, compensation payment, and evidence submission, with each step depicted in Figure 8. If the seller S_i has a problem delivering the auctioned item and wants to revoke the transaction, it proposes to recall. First, S_i computes a reasonable compensation c_{ij} for the buyer B_j , which consists of two parts: utility loss and cost offset, as follows:

$$c_{ij} = \mu \cdot (bid_i - p_{ij}) + v \cdot \omega \cdot \frac{t_{matching}}{\Delta},$$

where μ and v are the evaluation factors for the utility of winning the auction and cost in the matching process, and $\mu + v = 1$. The first term in the equation is the utility of seller i obtained by executing the trading. We use the seller’s utility here to delegate the buyer’s, as the compensation is computed by the seller who is ignorant of the buyer’s true bid. The second term denotes the expected cost generated from the previous stage of computing the allocation result, where ω is the expected unit cost of a user during the commitment, bidding, and allocation processes in the matching, and $t_{matching}$ represents the online time for the buyer during the matching. Then, S_i sends the revocation request including the amount of compensation to B_j . On receiving the revocation request, the buyer B_j can decide whether to agree with the revocation or not for his own sake. If B_j agrees to the revocation, it sends an agreement message to S_i and waits for the compensation payment. If the buyer wants more compensation than S_i has provided or insists on charging, it sends back a disagreement message. In the first case of agreement, S_i transfers the compensation to B_j in the channel. The revocation is completed in one round. The evidence is packed and sent to the smart contract for verification as Algorithm 1.

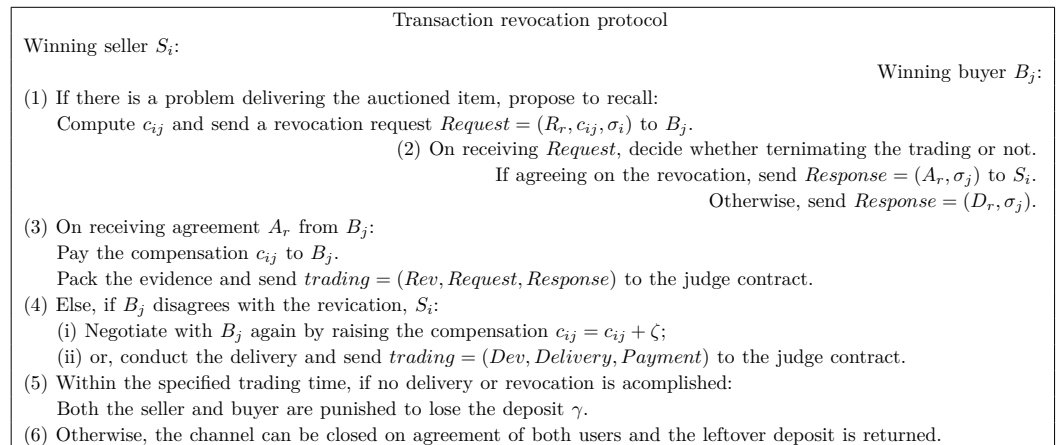


Figure 8. The transaction revocation protocol.

In the case of a one-round negotiation, the successful rate of revocation is quite low if the buyer disagrees. Otherwise, it is extremely inefficient if the negotiation continues without limitation. To balance revocation efficiency and success ratio, we use two-round negotiation in PAS, which is depicted in Figure 9. In the case of B_j 's disagreement, S_i has two normal choices. First, it can just choose to deliver the auctioned item and wait for B_j 's payment to accomplish the trading using fair exchange protocol [46]. Second, it can choose to negotiate with the buyer again by raising the amount of compensation $c_{ij} = c_{ij} + \zeta$, where ζ is the compensation increase that S_i can accept to realize the revocation. In theory, the negotiation between S_i and B_j can keep going on until B_j agrees with the revocation. But considering the time constraints and the cost of transaction revocation, PAS allows at most two rounds of negotiation. If B_j still disagrees with the revocation in two rounds, S_i must deliver the auctioned item. If the transaction revocation is successfully conducted in time, S_i packs and sends all the evidence to the judge contract, which reinforces the revocation verification by executing Algorithm 1. If the proofs are verified, the channel can be closed on the agreement of both users and the remained deposit will be returned. Otherwise, if some user violates the protocol, the deposit will be deducted and sent to the other user for compensation. Note that seller aborting is not a choice when performing the negotiation for two reasons. First, the seller is not near energy depletion and can not perform the negotiation, as the winning seller in the matching has enough energy to charge the buyer. Second, aborting will result in losing deposit γ , which is a severe loss that a rational seller won't choose.

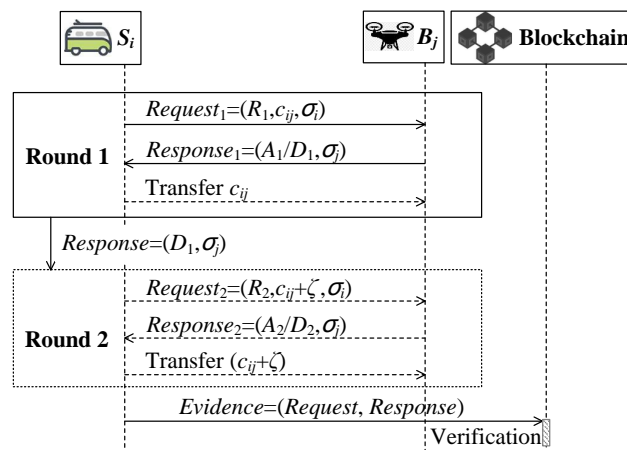


Figure 9. The rounds of revocation negotiation.

5. Theoretical Analysis

As aforementioned, security and privacy are important concerns in the open energy trading environment. We now prove that our proposed scheme is secure in the UC framework. We use the definition of UC-security from [31] to prove the designed off-chain protocol UC realizes the ideal functionality. We adopt the time consistency assumption from [41] for both real and ideal worlds. To prove PAS is UC-secure, we need to prove (1) the security of the on-chain and off-chain connection (Lemma 1); (2) the off-chain protocols UC-realize \mathcal{F}_{PAS} (Theorem 1). In other words, we need to construct different simulators \mathcal{S} in the ideal world. The function $Submit()$ is the on-chain and off-chain connection, as it is called by both the charging scheduling protocol and transaction revocation protocol. Hence, we first define $\mathcal{S}_{Submit}()$ as the simulator of $Submit()$ to prove that the view of environment \mathcal{E} remains the same in both the ideal world and the $(\mathcal{F}_{Judge}, \mathcal{F}_{\mathcal{L}})$ -hybrid world.

Lemma 1. We can define $\mathcal{S}_{Submit}()$ as the simulator of $Submit()$ such that the view of any environment \mathcal{E} remains the same in both the ideal world and the $(\mathcal{F}_{Judge}, \mathcal{F}_{\mathcal{L}})$ -hybrid world under the time consistency assumption.

Proof of Lemma 1. For matching results, if the user U_i calling $Submit()$ is corrupted, on U_i sending $Submit(matched(U_w, Pairs))$ to \mathcal{F}_{Judge} :

- (1) $\mathcal{S}_{Submit}()$ waits Δ rounds for other matching result submission. Then verify the signature, if $Ver(\sigma_i) = \perp$, stop.
- (2) If $Ver(\sigma_i) = \top$, $\mathcal{S}_{Submit}()$ waits $(n - 2)\Delta$ rounds for other matching result submission.
- (3) If all other users $U_{j \neq i}$ send $Submit(matched(U'_w, Pairs'))$ to \mathcal{F}_{Judge} then instruct \mathcal{F}_{PAS} to accept $(U'_w, Pairs')$ as the matching result.

If $U_{j \neq i}$ is corrupted, $\mathcal{S}_{Submit}()$ accept $(U_w, Pairs)$ as the matching result in the same round as the real world.

For trading result, if U_i is corrupted, on U_i sending $Submit(trading(Rev, Evidence))$ to \mathcal{F}_{Judge} :

- (1) $\mathcal{S}_{Submit}()$ waits Δ rounds for other trading result submission. Then verify the signature, if $Ver(\sigma_i) = \perp$, stop; else, verify the evidence $Evidence$ and confirm the revocation.
- (2) If the other user $U_{j \neq i}$ send $Submit(trading(Dev, Evidence))$ to \mathcal{F}_{Judge} then verify the evidence and confirm the trading result.

If $U_{j \neq i}$ is corrupted, $\mathcal{S}_{Submit}()$ can still confirm the trading result in the same round as the real world.

Since $Submit()$ can change the on-chain state of \mathcal{F}_{Judge} , the $\mathcal{S}_{Submit}()$ ensures that \mathcal{F}_{PAS} also accepts the same matching result and confirms the same trading result in the same round. Thus, the views of the ideal world and the $(\mathcal{F}_{Judge}, \mathcal{F}_L)$ -hybrid world are consistent. \square

Theorem 1. The designed off-chain protocols UC-realize \mathcal{F}_{PAS} in the $(\mathcal{F}_{Judge}, \mathcal{F}_L)$ -hybrid model under the time consistency assumption.

Proof of Theorem 1. We define the simulator \mathcal{S} for each functionality except the open multi-party channel, which can be referred to in [41].

Open two-party channel. If the channel initiator S_i is corrupted, on S_i sending $create()$ to \mathcal{F}_{Judge} :

- (1) \mathcal{S} sends $create()$ to \mathcal{F}_{PAS} to make sure that \mathcal{F}_{PAS} receives $create()$ in the same round as \mathcal{F}_{Judge} .
- (2) Wait Δ rounds for B_j .
- (3) Send $created()$ to \mathcal{E} if $\mathcal{F}_{PAS}(channel2) = created$.

If B_j is corrupted, on S_i sending $create()$ to \mathcal{F}_{PAS} :

- (1) \mathcal{S} waits Δ rounds for B_j .
- (2) Send $create()$ to \mathcal{F}_{PAS} if B_j sends $create()$ to \mathcal{F}_{Judge} .
- (3) Send $created()$ to \mathcal{E} if $\mathcal{F}_{PAS}(channel2) = created$.

In both cases, S_i and B_j output $created()$ to \mathcal{E} if $\mathcal{F}_{PAS}(channel2) = created$, while \mathcal{S} outputs $created()$ in the same round, i.e., \mathcal{E} receives the same outputs in both worlds in the same round.

Close multi-party channel. If the closure initiator U_i is corrupted, on U_i sending $close()$ to \mathcal{F}_{Judge} :

- (1) \mathcal{S} waits Δ rounds if $\mathcal{F}_{PAS}(channel2) = created$. Otherwise, stop.
- (2) Send $close()$ to \mathcal{F}_{PAS} to make sure \mathcal{F}_{PAS} receives $close()$ in the same round as \mathcal{F}_{Judge} , then wait $1 + (n - 1)\Delta$ rounds for U_j .
- (3) Check if $\mathcal{F}_{PAS}(channel1) = \perp$, send $closed()$ to \mathcal{E} on behalf of U_i .

If $U_{j \neq i}$ is corrupted, on U_i sending $close()$ to \mathcal{F}_{PAS} :

- (1) \mathcal{S} sends $close()$ to \mathcal{F}_{PAS} .
- (2) Wait $(n - 2)\Delta$ rounds for U_k .

- (3) Check if $\mathcal{F}_{PAS}(channel1) = \perp$, send $closed()$ to \mathcal{E} on behalf of U_j .

The indistinguishability in the view of \mathcal{E} between the two worlds holds in the same manner.

Close two-party channel. If the closure initiator S_i is corrupted, on S_i sending $close()$ to \mathcal{F}_{Judge} :

- (1) \mathcal{S} sends $close()$ to \mathcal{F}_{PAS} to make sure that \mathcal{F}_{PAS} receives $close()$ in the same round as \mathcal{F}_{Judge} .
- (2) Wait Δ rounds for B_j .
- (3) Check if $\mathcal{F}_{PAS}(channel2) = \perp$, send $closed()$ to \mathcal{E} on behalf of U_i .

If B_j is corrupted, on S_i sending $close()$ to \mathcal{F}_{PAS} :

- (1) \mathcal{S} sends $close()$ to \mathcal{F}_{PAS} .
- (2) Wait 1 round.
- (3) If $\mathcal{F}_{PAS}(channel2) = \perp$, send $closed()$ to \mathcal{E} on behalf of B_j .

\mathcal{E} receives the same outputs in both worlds in the same round. The indistinguishability holds.

Trade revocation. If S_i is corrupted, on S_i sending $Request(R_1, c_{ij}, \sigma_i)$:

- (1) \mathcal{S} sends $Request(R_1, c_{ij}, \sigma_i)$ to \mathcal{F}_{PAS} .
- (2) If B_j sends $Response(A_1, \sigma_j)$, \mathcal{S} sends $Response(A_1, \sigma_j)$ in the same round.
- (3) Otherwise, if B_j sends $Response(D_1, \sigma_j)$ then \mathcal{S} sends $Response(A_1, \sigma_j)$ to \mathcal{F}_{PAS} in the same round.
- (4) If S_i executes $Transfer()$, \mathcal{S} sends the $Transfer()$ in the same round. Else, if S_i executes $Request(R_2, c_{ij} + \zeta, \sigma_i)$, \mathcal{S} sends $Request(R_2, c_{ij} + \zeta, \sigma_i)$ in the same round.
- (5) If S_i executes $Submit(Trading(type, Evidence))$ to \mathcal{F}_{Judge} , \mathcal{S} calls $S_Submit(Trading(type, Evidence))$ to make sure that \mathcal{F}_{PAS} receives $Trading(type, Evidence)$ in the same round as \mathcal{F}_{Judge} .

If B_j is corrupted, on S_i sending $Request(R_1, c_{ij}, \sigma_i)$ to \mathcal{F}_{PAS} :

- (1) \mathcal{S} forwards $Request(R_1, c_{ij}, \sigma_i)$ to \mathcal{F}_{PAS} .
- (2) If B_j sends $Response(A_1, \sigma_j)$, \mathcal{S} sends $Response(A_1, \sigma_j)$ to \mathcal{F}_{PAS} . Else, if B_j sends $Response(D_1, \sigma_j)$, \mathcal{S} sends $Response(D_1, \sigma_j)$ to \mathcal{F}_{PAS} .
- (3) If S_i executes $Transfer()$, \mathcal{S} sends $Transfer()$ to \mathcal{F}_{PAS} in the same round. If S_i executes $Request(R_2, c_{ij} + \zeta, \sigma_i)$, \mathcal{S} sends $Request(R_2, c_{ij} + \zeta, \sigma_i)$ in the same round.
- (4) If S_i executes $Submit(Trading(type, Evidence))$, \mathcal{S} calls $S_Submit(Trading(type, Evidence))$ in the same round.
- (5) If B_j sends $Submit(Trading(type, Evidence))$ to \mathcal{F}_{Judge} , \mathcal{S} calls $S_Submit(Trading(type, Evidence))$ in the same round.

Since the messages exchanged between any entities are the same in both worlds, the view of \mathcal{E} between the two worlds is indistinguishable, i.e., the indistinguishability holds.

Trade matching. Suppose U_i is the user computing the matching result. If U_i is corrupted, on U_i broadcasting C_i to other users:

- (1) \mathcal{S} sends C_i to \mathcal{F}_{PAS} .
- (2) If \mathcal{F}_{PAS} doesn't receive all commitments, stop. Else if U_i executes $Submit()$, \mathcal{S} calls $S_Submit()$ in the same round.
- (3) If U_i broadcasts b_i , \mathcal{S} sends b_i to \mathcal{F}_{PAS} ; else, stop.
- (4) If \mathcal{F}_{PAS} doesn't receive all bids, stop. Else if U_i executes the $Submit()$, \mathcal{S} calls $S_Submit()$ in the same round.

If $U_{j \neq i}$ is corrupted, on U_i sending $matching()$ to $\mathcal{F}_{Auction}$:

- (1) If U_i sends C_i to \mathcal{F}_{PAS} , \mathcal{S} forwards C_i in the same round.
- (2) If U_j broadcasts C_j , \mathcal{S} sends C_j to \mathcal{F}_{PAS} ; else, \mathcal{S} executes $S_Submit()$ to report U_j refuses to broadcast and stop.
- (3) If U_i sends b_i to \mathcal{F}_{PAS} , \mathcal{S} forwards b_i to U_j in the same round; else, stop.

- (4) If U_j broadcasts b_j , S sends b_j to \mathcal{F}_{PAS} ; else, S executes $S_Submit()$ to report U_j refuses bidding and stop.
- (5) If U_j executes $Submit()$, S calls $S_Submit()$ in the same round; else, stop.

All messages exchanged between any entities are exactly the same in both worlds, so the indistinguishability holds. \square

6. Performance Analysis

In this section, we conduct an experimental analysis to evaluate the performance of the proposed scheme. We implemented the logic procedure of PAS under our developed framework based on Ethereum. The smart contract was written in Solidity 0.4.24. It is developed in the Truffle framework and then deployed on a locally simulated network Ganache. We used a laptop with 1.6 GHz Dual-Core Intel Core i5 CPU and 8 GB of memory. In Ethereum, transaction fees are paid by the transaction sender to the miners. The fees are calculated using “gas”, which is a crucial evaluation criterion when using blockchain contracts. The gas consumption scheme is not suitable for UAV charging. We first test the gas cost to deploy the PAS contract. Then gas consumption for trade matching and trade revocation is evaluated and compared with the on-chain mechanism. Moreover, the calculation amount and the interaction complexity are also important efficiency issues, as UAVs are energy constrained. So we test the revocation latency, which reflects the intricacy of the computation and interaction.

In our framework, the procedures of channel creation and closure, matching result submission, and revocation verification will cause on-chain costs. The charging scheduling and revoking negotiation are conducted off-chain for privacy preservation and on-chain computation cost saving. We assume that, in the beginning, all users collect each other’s public key. In addition, every user creates an Ethereum account. The gas amount needed for a transaction is related to two factors—the amount of data and the computational effort. We need to pay about 1,019,957 gas to deploy the judge contract. We implement all the functionalities in a single smart contract. Our implementation goal is not to optimize the deployment cost. To mitigate the deployment cost, the process could transfer every functionality of the judge contract into an external library. In the following, we ignore the deployment cost when assessing PAS.

For the experiments, we follow the setup of a numerical experiment consisting of 110 users in total, where 70 are buyers and 40 are sellers. All users accomplish the charging scheduling in a multi-party state channel based on a secure double auction, while the revocation or trading is conducted in a two-party state channel established by each pair of winning users. If all users unanimously agree to create the multi-party state channel, each user sends one transaction to the smart contract. In all, 110 on-chain transactions are issued, which costs about 6,447,210 gas as shown in Table 1. To close the multi-party channel, there are 110 more transactions that cost a little less. This is because the deposit is deducted from each user’s account during the channel creation. The whole scheduling process is executed off-chain, so no gas is consumed. For trading and revoking between the winning buyer and seller, only two on-chain transactions are sent to create and close the two-party channel, with each user consuming about 58,611 and 54,704 gas for creation and closure, respectively. The revoking negotiation and trading are performed off-chain without any gas cost.

Table 1. The gas costs and message complexity with 110 users.

	On-Chain Tx	Gas (a User/Total)	Off-Chain Message
Create multi-party channel	110	58,611/6,447,210	0
Charging scheduling	0	0	4×110
Close multi-party channel	110	54,704/6,017,440	0
Create two-party channel	2	58,611/117,222	0
Transaction revocation	0	0	3/6
Close two-party channel	2	54,704/109,408	0

To illustrate the scalability of PAS, we test the gas consumption with the increasing numbers of users. Figure 10 shows the cumulative gas cost with different numbers of users. We use the on-chain trade matching as the baseline, which is gas-consuming because all computations are conducted by all miners on the chain. We can observe that the gas cost for charging scheduling and revoking grows almost linearly with the user number if there are no state channels. This is because each user sends on-chain transactions to the Judge contract to commit and reveal the bid, request for revoking, and agree or disagree with the revoking. On the contrary, the gas cost of PAS increases placidly for charging scheduling with the user number rising. As the number of users increases, the additional on-chain cost of PAS is just related to the multi-party channel creation and closure. It almost keeps the same for revoking a transaction for each revocation is performed in different channels. For charging scheduling, two more on-chain transactions are needed with a new user joining to create or close the multi-party channel. As a result, the proposed protocol reduces the amount of on-chain transactions and computations. PAS thus achieves cost savings and scalability. To achieve secure attestation of the revocation, the revocation verification is conducted on-chain by the Judge contract, where the cost differs according to the trading type of delivery or revocation. The number of off-chain messages transferred during transaction revocation varies with one-round and two-round negotiation according to the designed two-round negotiation protocol. Table 1 summarizes the on-chain and off-chain costs of PAS' execution.

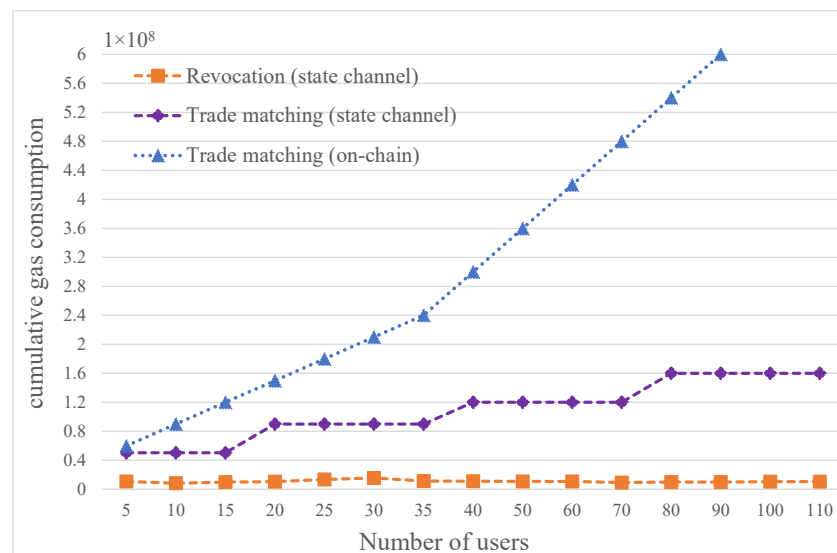


Figure 10. The cumulative gas consumption with the number of users.

To better capture the performance of our system under transaction revocation, Figure 11 illustrates the end-to-end delay between the seller and buyer with one and two rounds of revoking negotiation. Creating and closing the two-party channel would take about 206 ms and 177 ms, respectively. The end-to-end latency is dominated by the time it takes to negotiate between the seller and the buyer off-chain. As can be seen from Figure 11a, the latency increases if the buyer disagrees with the revoking in the first round. Because the seller may proceed to the second round of negotiation by raising the compensation. The average latency increases linearly with the number of users as more winning pairs may propose to revoke with a rising number of users, as shown in Figure 11b. The designed two-round negotiation protocol achieves an acceptable trade-off between efficiency and success ratio of transaction revocation.

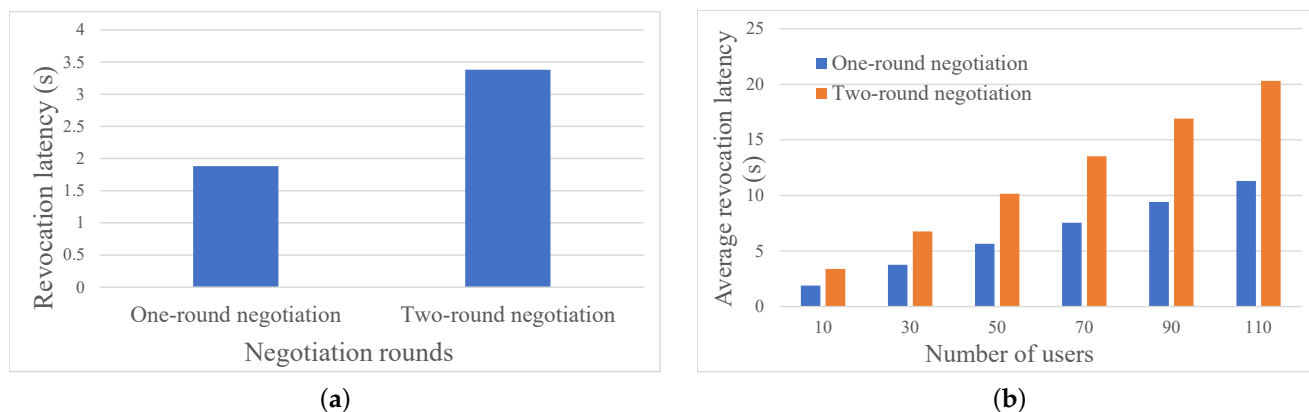


Figure 11. The revocation latency. (a) Latency with different negotiation rounds. (b) Average latency with the number of users.

In general, the implementation and experiments have demonstrated that the proposed PAS is feasible and practical for UGVs-assisted UAV charging. Compared to the on-chain method, our scheme achieves both computational efficiency and cost savings with a security and fairness guarantee. Furthermore, as PAS induces a relatively small overhead, it is able to support a large number of participating users, which is in accordance with the real-world IoT application scenario.

7. Conclusions

To realize reliable revocation attestation with privacy preservation in blockchain-based IoT applications, suitable for UGVs-assisted UAV charging, a trust-free scheme is proposed by using hybrid state channels, combining reliable on-chain verification and attestation with private and efficient off-chain computation. To address the issue of privacy leakage derived from the transparency of the blockchain, different channels and distributed encryption are adopted to realize lightweight privacy preservation. To overcome the unfairness in transaction revocation, a fair negotiation and compensation protocol is proposed, where users perform the revocation negotiation through a two-party state channel for efficiency and privacy. The analyses prove the security properties of our design with both on-chain and off-chain operations using the simulation-based UC framework. The experiments conducted demonstrate the feasibility and practicality of our proposed scheme.

Author Contributions: X.J. designed the system framework and realized and verified the proposed scheme. X.S. was responsible for the determination of the methodology and the organizational structure. C.Y. revised the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Key R&D Program of China (No. 2020YFB1005500), the Leading-edge Technology Program of Jiangsu Natural Science Foundation (No. BK20202001), and the Postdoctoral Science Foundation of Jiangsu Province (No. 2021K596C).

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declared no potential conflict of interest with respect to the research, authorship, or publication of this article.

References

1. Wang, Y.; Su, Z.; Zhang, N.; Li, R. Mobile wireless rechargeable UAV networks: Challenges and solutions. *IEEE Commun. Mag.* **2022**, *60*, 33–39. [[CrossRef](#)]
2. Su, Z.; Wang, Y.; Xu, Q.; Zhang, N. Lvbs: Lightweight vehicular blockchain for secure data sharing in disaster rescue. *IEEE Trans. Depend. Secure Comput.* **2020**, *19*, 19–32. [[CrossRef](#)]
3. Shi, M.; Feng, X.; Pan, S.; Song, X.; Jiang, L. A collaborative path planning method for intelligent agricultural machinery based on unmanned aerial vehicles. *Electronics* **2023**, *12*, 3232. [[CrossRef](#)]

4. Lakew, D.S.; Tran, A.T.; Dao, N.N.; Cho, S. Intelligent offloading and resource allocation in heterogeneous aerial access internet networks. *IEEE Internet Things J.* **2023**, *10*, 5704–5718. [[CrossRef](#)]
5. Mozaffari, M.; Saad, W.; Bennis, M.; Debbah, M. Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 7574–7589. [[CrossRef](#)]
6. Mohsan, S.A.H.; Khan, M.A.; Noor, F.; Ullah, I.; Alsharif, M.H. Towards the unmanned aerial vehicles (UAVs): A comprehensive review. *Drones* **2022**, *6*, 147. [[CrossRef](#)]
7. Mohsan, S.A.H.; Othman, N.Q.H.; Khan, M.A.; Amjad, H.; Żywiołek, J. A comprehensive review of micro UAV charging techniques. *Micromachines* **2022**, *13*, 977. [[CrossRef](#)]
8. Wu, S.; Cai, C.; Liu, X.; Chai, W.; Yang, S. Compact and free-positioning omnidirectional wireless power transfer system for unmanned aerial vehicle charging applications. *IEEE Trans. Power Electron.* **2022**, *37*, 8790–8794. [[CrossRef](#)]
9. Wu, M.; Su, L.; Chen, J.; Duan, X.; Wu, D.; Cheng, Y.; Jiang, Y. Development and prospect of wireless power transfer technology used to power unmanned aerial vehicle. *Electronics* **2022**, *11*, 2297. [[CrossRef](#)]
10. Li, M.; Liu, L.; Gu, Y.; Ding, Y.; Wang, L. Minimizing energy consumption in wireless rechargeable UAV networks. *IEEE Internet Things J.* **2021**, *9*, 3522–3532. [[CrossRef](#)]
11. Shin, M.; Kim, J.; Levorato, M. Auction-based charging scheduling with deep learning framework for multi-drone networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 4235–4248. [[CrossRef](#)]
12. Xu, J.; Zhu, K.; Wang, R. Rf aerially charging scheduling for UAV fleet: A q-learning approach. In Proceedings of the 2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), Shenzhen, China, 11–13 December 2019; pp. 194–199.
13. Ribeiro, R.G.; Cota, L.P.; Euzébio, T.A.; Ramírez, J.A.; Guimarães, F.G. Unmanned-aerial-vehicle routing problem with mobile charging stations for assisting search and rescue missions in postdisaster scenarios. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *52*, 6682–6696. [[CrossRef](#)]
14. Guo, J.; Liu, Z.; Tian, S.; Huang, F.; Li, J.; Li, X.; Igorevich, K.K.; Ma, J. Tfl-dt: A trust evaluation scheme for federated learning in digital twin for mobile networks. *IEEE J. Sel. Areas Commun.* **2023**, 1–14. [[CrossRef](#)]
15. Guo, J.; Ding, X.; Wu, W. A double auction for charging scheduling among vehicles using dag-blockchains. *arXiv* **2020**, arXiv:2010.01436.
16. Jia, X.; Song, X.; Sohail, M. Effective consensus-based distributed auction scheme for secure data sharing in internet of things. *Symmetry* **2022**, *14*, 1664. [[CrossRef](#)]
17. Yu, C.; Zhan, Y.; Sohail, M. SDSM: Secure data sharing for multilevel partnerships in IoT based supply chain. *Symmetry* **2022**, *14*, 2656. [[CrossRef](#)]
18. Guo, J.; Ding, X.; Wang, T.; Jia, W. Combinatorial resources auction in decentralized edge-thing systems using blockchain and differential privacy. *Inf. Sci.* **2022**, *607*, 211–229. [[CrossRef](#)]
19. Dunphy, P.; Petitcolas, F.A. A first look at identity management schemes on the blockchain. *IEEE Secur. Priv.* **2018**, *16*, 20–29. [[CrossRef](#)]
20. Rosa, R.V.; Rothenberg, C.E. Blockchain-based decentralized applications for multiple administrative domain networking. *IEEE Commun. Stand. Mag.* **2018**, *2*, 29–37. [[CrossRef](#)]
21. Hassija, V.; Chamola, V.; Krishna, D.N.G.; Guizani, M. A distributed framework for energy trading between UAVs and charging stations for critical applications. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5391–5402. [[CrossRef](#)]
22. Torky, M.; El-Dosuky, M.; Goda, E.; Snášel, V.; Hassani, A.E. Scheduling and securing drone charging system using particle swarm optimization and blockchain technology. *Drones* **2022**, *6*, 237. [[CrossRef](#)]
23. Ma, J.; Xu, S.; Ning, J.; Huang, X.; Deng, R.H. Redactable blockchain in decentralized setting. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 1227–1242. [[CrossRef](#)]
24. Panwar, G.; Vishwanathan, R.; Misra, S. Retrace: Revocable and traceable blockchain rewrites using attribute-based cryptosystems. In Proceedings of the 26th ACM Symposium on Access Control Models and Technologies, Virtual Event, 16–18 June 2021; pp. 103–114.
25. Gates, V. Rtm: Blockchain that support revocable transaction model. *arXiv* **2020**, arXiv:2001.11259.
26. Huang, K.; Zhang, X.; Mu, Y.; Wang, X.; Yang, G.; Du, X.; Rezaeibagha, F.; Xia, Q.; Guizani, M. Building redactable consortium blockchain for industrial internet-of-things. *IEEE Trans. Ind. Inform.* **2019**, *15*, 3670–3679. [[CrossRef](#)]
27. Wu, G.; Ren, P.; Du, Q. Recall-based dynamic spectrum auction with the protection of primary users. *IEEE J. Sel. Areas Commun.* **2012**, *30*, 2070–2081. [[CrossRef](#)]
28. Yi, C.; Cai, J. Two-stage spectrum sharing with combinatorial auction and stackelberg game in recall-based cognitive radio networks. *IEEE Trans. Commun.* **2014**, *62*, 3740–3752. [[CrossRef](#)]
29. Yi, C.; Cai, J. Multi-item spectrum auction for recall-based cognitive radio networks with multiple heterogeneous secondary users. *IEEE Trans. Veh. Technol.* **2015**, *64*, 781–792. [[CrossRef](#)]
30. Liu, M.; Wu, Q.; Hei, Y.; Li, D.; Hu, J. Fair and smart spectrum allocation scheme for IIoT based on blockchain. *Ad Hoc Netw.* **2021**, *123*, 102686. [[CrossRef](#)]
31. Malavolta, G.; Moreno-Sanchez, P.; Kate, A.; Maffei, M.; Ravi, S. Concurrency and privacy with payment-channel networks. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 455–471.

32. Khalil, R.; Gervais, A. Revive: Rebalancing off-blockchain payment networks. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 439–453.
33. Dziembowski, S.; Eckey, L.; Faust, S.; Malinowski, D. Perun: Virtual payment hubs over cryptocurrencies. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 106–123.
34. Miller, A.; Bentov, I.; Bakshi, S.; Kumaresan, R.; McCorry, P. Sprites and state channels: Payment networks that go faster than lightning. In Proceedings of the International Conference on Financial Cryptography and Data Security, St. Kitts, St. Kitts and Nevis, 18–22 February 2019; pp. 508–526.
35. Lei, H.; Huang, L.; Wang, L.; Chen, J. Mpc: Multi-node payment channel for off-chain transactions. In Proceedings of the ICC 2022-IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 4733–4738.
36. Chen, Y.; Li, X.; Zhang, J.; Bi, H. Multi-party payment channel network based on smart contract. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 4847–4857. [[CrossRef](#)]
37. Dziembowski, S.; Faust, S.; Hostáková, K. General state channel networks. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 949–966.
38. Close, T.; Stewart, A. *Forcemove: An n-Party State Channel Protocol*; White Paper; Magmo: London, UK, 2018.
39. McCorry, P.; Buckland, C.; Bakshi, S.; Wüst, K.; Miller, A. You sank my battleship! A case study to evaluate state channels as a scaling solution for cryptocurrencies. In Proceedings of the International Conference on Financial Cryptography and Data Security, St. Kitts, St. Kitts and Nevis, 18–22 February 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 35–49.
40. Buckland, C.; McCorry, P. Two-party state channels with assertions. In Proceedings of the International Conference on Financial Cryptography and Data Security, St. Kitts, St. Kitts and Nevis, 18–22 February 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 3–11.
41. Nguyen, T.D.; Thai, M.T. A blockchain-based iterative double auction protocol using multi-party state channels. *ACM Trans. Internet Technol.* **2021**, *21*, 39. [[CrossRef](#)]
42. Abe, M.; Suzuki, K. M+ 1-st price auction using homomorphic encryption. In Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems (PKC), Paris, France, 12–14 February 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 115–124.
43. Wang, C.; Leung, H.F.; Wang, Y. Secure double auction protocols with full privacy protection. In Proceedings of the 6th International Conference on Information Security and Cryptology (ICISC), Seoul, Korea, 27–28 November 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 215–229.
44. Pedersen, T.P. Non-interactive and information-theoretic secure verifiable secret sharing. In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 11–15 August 1991; Springer: Berlin/Heidelberg, Germany, 1991; pp. 129–140.
45. McAfee, R.P. A dominant strategy double auction. *J. Econ. Theory* **1992**, *56*, 434–450. [[CrossRef](#)]
46. Dziembowski, S.; Eckey, L.; Faust, S. Fairswap: How to fairly exchange digital goods. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 967–984.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.