Onchain Auction: A Trustless Protocol for Permissionless Price Discovery

Otoshi
https://onchain-auction.com

Abstract

We present a fully decentralized smart contract protocol for permissionless English auctions of ERC-20 tokens and ERC-721 non-fungible tokens on Ethereum. The protocol uses a pull-payment architecture for all ETH disbursements, eliminating a well-known class of griefing attacks in which a non-payable participant blocks other users from participating. An anti-snipe time extension mechanism prevents last-second bidding from suppressing honest price discovery. The contract has no owner, no administrator, no upgrade mechanism, and no pause function.

1. Introduction

Auctions are among the oldest mechanisms for price discovery. From Roman property sales to modern spectrum allocation, the English ascending-price auction has endured because it efficiently aggregates dispersed information about an asset's value into a single price through competitive bidding.

The digitization of auctions introduced a new category of infrastructure risk. Centralized auction platforms operate as trusted intermediaries: they custody assets during the auction period, process bids, determine winners, and disburse proceeds. This centralization creates three failure modes not present in the auction mechanism itself:

Platform risk. The operator may cease operations, suffer a security breach, or become subject to regulatory action that prevents settlement. Assets and proceeds held in custody may be frozen or lost.

Censorship risk. The operator retains discretionary authority over participation. Sellers may be deplatformed, assets may be delisted, and bidders may be excluded based on criteria unrelated to the auction itself.

Custody risk. During the auction period, the seller's asset and the bidders' capital are held by the operator rather than by the participants or a neutral contract. The operator becomes a single point of failure for all assets in custody.

This paper presents Onchain Auction, a protocol that implements English auctions on Ethereum without any centralized operator. The protocol addresses a fourth risk category specific to on-chain auctions: griefing attacks enabled by the interaction between Ethereum's execution model and naive payment patterns.

2. The Pull-Payment Pattern

The most critical design decision in this protocol is the use of pull-payment for all ETH disbursements. To understand why this is necessary, consider the standard push-payment approach used by many on-chain auction implementations.

In a push-payment auction, when a new bid is placed, the contract immediately attempts to refund the previous highest bidder by sending ETH to their address. If the previous bidder is an externally owned account (EOA), this transfer succeeds. However, if the previous bidder is a smart contract whose receive() or fallback() function reverts, the refund fails, and because the refund is part of the bid transaction, the entire bid transaction reverts.

This creates a griefing attack: a malicious actor deploys a contract that always reverts on ETH receipt, uses it to place a bid, and thereby prevents all subsequent bids from being processed. The auction is effectively frozen, and the attacker may win at an artificially low price.

The same vulnerability affects settlement. If the winning bidder, the seller, or the fee recipient is a non-payable contract, the settlement transaction reverts and the auction cannot be concluded. Assets and funds may become permanently locked.

Onchain Auction eliminates this entire class of attacks through the pull-payment pattern. Instead of sending ETH directly to recipients, the contract credits their balance in a pendingWithdrawals mapping. Recipients call withdraw() to claim their credited balance at any time. The critical property is that no external address can cause the bid() or settle() functions to revert due to a failed ETH transfer.

2.1 Formal Properties

The pull-payment architecture guarantees three invariants:

Bid liveness. For any active auction with time remaining, any transaction calling bid() with a valid bid amount will succeed, regardless of the identity or implementation of any previous bidder.

Settlement liveness. For any ended auction, any transaction calling settle() will succeed, regardless of the identity or implementation of the winning bidder, the seller, or the fee recipient.

Refund availability. Any address with a non-zero pendingWithdrawals balance can claim that balance by calling withdraw(), subject only to the gas requirements of a simple ETH transfer.

3. Anti-Snipe Time Extension

Sniping is the practice of placing a bid in the final seconds of an auction to win at a price below the asset's true market value. In a physical auction, the auctioneer extends bidding whenever a new bid is placed. In a time-bounded online auction, the fixed end time creates an incentive to bid as late as possible, suppressing price discovery.

Onchain Auction implements a time extension mechanism: if a bid is placed within the last 10 minutes of an auction, the end time is extended by 10 minutes from the time of the bid. This ensures that all participants have a fair opportunity to respond to late bids, replicating the open-ended nature of physical auctions while

maintaining a bounded auction duration.

The 10-minute window and extension duration are hardcoded constants. They were chosen to provide sufficient time for a participant to observe a new bid on-chain, prepare a response transaction, and have that transaction included in a block, even under periods of moderate network congestion.

4. Auction Lifecycle

4.1 Creation

A seller creates an auction by depositing an ERC-20 token amount or an ERC-721 NFT into the contract and specifying five parameters: starting price (minimum first bid), reserve price (minimum acceptable winning bid, or zero for no reserve), minimum bid increment, and duration. For ERC-20 tokens, the contract measures the actual amount received to accommodate transfer tax tokens.

4.2 Bidding

Bidders call bid() with ETH attached. The first bid must meet or exceed the starting price. Subsequent bids must exceed the current highest bid by at least the minimum increment. When a new highest bid is placed, the previous highest bidder's refund is credited to their pendingWithdrawals balance.

4.3 Settlement

After the auction end time (including any anti-snipe extensions), anyone may call settle(). If no bids were placed, the asset is returned to the seller. If the reserve price was not met, the asset is returned to the seller and the highest bidder's refund is credited. If the reserve was met, the asset is transferred to the winning bidder, the seller's proceeds are credited, and the protocol fee is credited to the fee recipient.

4.4 Cancellation

The seller may cancel an auction that has received no bids. Once a bid has been placed, cancellation is no longer available, protecting bidders from sellers who cancel after seeing insufficient interest.

5. Multi-Asset Support

The contract supports two asset standards. ERC-20 tokens are transferred using OpenZeppelin's SafeERC20, with balance measurement accommodating non-standard transfer implementations. ERC-721 non-fungible tokens are transferred using safeTransferFrom, with the contract inheriting ERC721Holder to accept incoming NFTs. The contract validates NFT ownership upon deposit and transfers ownership to the winning bidder upon settlement.

6. Security Architecture

All state-changing external functions are protected by ReentrancyGuard. ERC-20 transfers use SafeERC20. ERC-721 custody uses ERC721Holder. All ETH disbursements use the pull-payment pattern with Checks-Effects-Interactions ordering. Existence guards validate auction IDs before all operations.

The contract contains no selfdestruct instruction, no delegatecall, no external dependencies, and no administrative functions. Its behavior is fully determined by its bytecode and the state of the Ethereum blockchain.

7. The Case for Immutability

An upgradeable contract has a strictly larger attack surface than an immutable one. Every upgrade mechanism requires at least one privileged address with authority to modify the contract's behavior. This address is a permanent vulnerability. Governance mechanisms distribute but do not eliminate this risk.

For auction infrastructure, immutability provides a specific additional benefit: it guarantees that the rules of the auction cannot be changed after a seller has deposited an asset or a bidder has placed a bid. An upgradeable auction contract could, in principle, be modified mid-auction to change the fee structure, modify the settlement logic, or redirect proceeds. An immutable contract cannot.

The strongest test of a protocol's decentralization is whether it would continue to function if its creators ceased to exist. This contract passes that test.

8. Contract

```
Address: 0x533BaD2ea0B0F343bDe6a8dc89b887257561A35a
Chain: Ethereum Mainnet (Chain ID: 1)
Owner: None
Compiler: Solidity ^0.8.24, Optimizer 200 runs
```

9. Conclusion

Onchain Auction demonstrates that permissionless price discovery can be implemented without platform risk, censorship risk, custody risk, or griefing risk. The pull-payment architecture is a meaningful and necessary security improvement over push-payment designs and should be considered a best practice for any on-chain protocol that disburses ETH to addresses outside its direct control.

The anti-snipe mechanism ensures that price discovery is not suppressed by strategic timing. The permissionless settlement ensures that auction outcomes are finalized without reliance on any specific party. The contract is permanent auction infrastructure on Ethereum.

References

[1] EIP-20: Token Standard. Ethereum Improvement Proposals, 2015.

[2] EIP-721: Non-Fungible Token Standard. Ethereum Improvement Proposals, 2018.

[3] OpenZeppelin Contracts. Security library for Solidity smart contracts.

[4] V. Krishna, "Auction Theory," Academic Press, 2002.

[5] C. Detrio, "Smart Contract Security Best Practices," ConsenSys, 2019.

— Otoshi

https://onchain-auction.com