TLTR

The recent announcement from OpenSea indicates that it is time to update the ERC721 protocol to enforce creator royalties at the protocol level. We need to move away from the current transaction model (which gives too much power to NFT marketplaces) to a trustless transaction mode.

I published my draft proposal, Decentralized NFT yesterday, and I attached the copy below.

Decentralized NFT

A major flaw of ERC721

While NFT is playing a very important role to bring more people into the Web3 ecosystem and even create new income opportunities for artists, the protocol itself, ERC721, is still very immature and has a few fundamental issues.

1. Royalties are not enforceable.
2. Many NFTs are stolen.
3. It is not decentralized.

Because royalties are not part of ERC721, it is up to the marketplaces, such as OpenSea, to pay royalties.

As the result, many royalty-free were born, and the “Race to the Bottom” has started.

A lot of NFTs are stolen by scam sites, which ask the user to connect his/her wallet, let him/her call ApprovalForAll (without understanding what it means), and steal NFTs.

The fundamental flaw of ERC721 is in this “approve & transfer” model, which gives too much power to the marketplace, far from the beauty of Web3’s “decentralized and trustless” model.

Once the token owner calls Approval or ApprovalForAll, the marketplace can do whatever they want to do to the token (or tokens).

This is why royalty-free marketplaces were born, and so many NFTs are stolen.

In order to solve this problem, we need to create a new mechanism, which performs transactions without trusted third parties (P2P transactions), and makes it possible to enforce royalties.

P2P/Trustless transactions

Decentralized transactions will become possible by adding three methods to ERC721.

interface ERC721P2P is ERC721 { function setPriceOf(uint256 _tokenId, uint256 _price) external; function getPriceOf(uint256 _tokenId) external view returns(uint256); function purchase(uint256 _tokenId, address _wallet) external payable; }

The token owner calls setPriceOf

method to set the asking price of a specific token he/she owns.

Anybody can access this asking price by calling the getPriceOf

method (it returns 0 if it is not specified).

Anybody can buy it by calling the purchase

method by paying the asking price.

The smart contract that receives this money (via purchase

method) distributes it between the token owner and the artist, enforcing the royalty payment agreement between them.

Please notice that this transaction happens just between the seller and the buyer, without involving any third party. Because of that, it is possible to make a trade by using Etherscan.

This is already a huge improvement over the current “approval & transfer” mode, but we need to solve another scenario where the buyer makes an offer first (or makes an offer lower than the asking price).

“Offer and Accept” scenario

There are a few possible ways to support this scenario, but I think the decentralized marketplace is the most reasonable solution.

interface IERC721Marketplace { function makeAnOffer(ERC721P2P _contract, uint256 _tokenId, uint256 _price) external payable; function withdrawAnOffer(ERC721P2P _contract, uint256 _tokenId) external; function getTheBestOffer(ERC721P2P _contract, uint256 _tokenId) external view returns(uint256, address); function acceptOffer(ERC721P2P _contract, uint256 _tokenId, uint256 _price) external; }

interface ERC721P2P is ERC721 { function setPriceOf(uint256 _tokenId, uint256 _price) external; function getPriceOf(uint256 _tokenId) external view returns(uint256); function purchase(uint256 _tokenId, address _wallet) external payable; function acceptOffer(uint256 _tokenId, IERC721Marketplace _dealer, uint256 _price) external; }

The buyer makes an offer by calling the makenAnOffer

method at an autonomous market place, staking the amount of money he/she is willing to pay.

The seller (or other buyers) can see the current best offer (on that particular marketplace) by calling the getTheBestOffer

method.

The buyer accepts this offer by calling the acceptOffer

method of the ERC721P2P contract, which matches the asking price to the offer price, and calls the acceptOffer

method of the specified autonomous marketplace.

The autonomous marketplace calls back the purhase

method with the money from the buyer, and let it complete the transaction.

ERC721 compatibility

For a smooth transition, it makes sense to make the new protocol compatible with ERC721, simply inheriting it. We, however, need to disable (or partially disable) the approval

and approvalForAll

method, in order to prevent transactions on royalty-free marketplaces.

During the transition period, it makes sense to have a whitelist of marketplaces, so that traditional “approval and transfer” style transactions can happen only on trusted marketplaces that pay royalties appropriately.

We don’t need to disable the transfer

method, which allows token owners to transfer their tokens to other wallets. We don’t need to eliminate off-the-market transactions completely.

Security Concerns

With this change, scam sites will attempt to let the user call the acceptOffer

method at a very low price. We certainly need a special UI on the wallet (such as Metamask), which presents the meaning of this transaction (the NFT and the offer price).