SOLIDITY FINANCE

MeDIA eYe - Smart Contract Audit Report

SUMMARY

MeDIA eYe is building an innovative Blockchain-as-a-Service platform that allows users to create, mint, explore, create events, distribute, and share NFTs and collections, while offering users the ability to sell, auction, buy NFTs in the marketplace and more.

We reviewed MeDIA eYe's contracts at commit 991cf0fe11cf9ce144c149629e296129b6690cc1 on the team's private GitHub repository.

Audit Findings Summary:

- *No security issues from outside attackers were identified.*
- *Please ensure trust in the project team as they have substantial power in the ecosystem.*
- *Date: October 29th, 2021.*

Notes on Individual Contracts:

MediaEyeFee Contract:

- *The MediaEyeFee contract is used to manage fees associated with the platform.*
- *Users must complete all transactions using only a payment token that is accepted by the platform; prices may vary based on the token used.*
- *Anyone can use this contract to pay for a Level One or Level Two Subscription for a period of 30 or 90 days; Subscriptions are valid across all the platform's supported chains.*
- *Anyone can use this contract to pay the upload fee, which varies based on the specified "Upload Tier".*
- *Anyone can use this contract to choose an eligible NFT to feature and must pay a feature fee to do so.*
- *As uploads and features are handled off chain, the users' payment data for uploads and features is not stored in the contract*

> - *Any Admin is able to grant any address a Level One or Level Two Subscription for any specified duration at any time.*
> - *Any Admin is able to update feature duration, the number of features per category, and how long in advance a feature must be booked at any time.*
> - *Any Admin is able to withdraw any ETH or ERC20 tokens from the contract at any time.*
> - *Any Admin is able to add or remove any token as an accepted payment token at any time.*
> - *Any Admin is able to set the upload fees, the subscription fees, and the feature fee for any accepted payment token at any time.*
> - *Any Admin is able to set the fee wallet to any address at any time.*
> - *Any Admin is able to set the mediator address to any address at any time, which is responsible for permeating subscription information across the platform's various supported chains.*
> - *The project team should exercise caution when adding accepted payment tokens as to not use fee-on-transfer tokens or ERC777-compliant tokens; if fee-on-transfer tokens are used, the team must ensure that this contract is excluded from transfer fees.*
>
> *MediaEyeHomeMediator, MediaEyeForeignMediator, and MediaEyeXdaiMediator Contracts:*
>
> - *The MediaEyeFee contract uses the MediaEyeHomeMediator contract to send subscription information from the home chain to the foreign chain.*

> - *The Mediator encodes the subscription information and uses the Bridge to deliver it to the xDai chain.*
> - *The data is stored on the xDai chain until invokeSubscribe() is called to deliver the data to the receiving chain via the Bridge.*
> - *The Admin is able to set the MediaEyeFee, xDaiMediator, and Bridge contract addresses to any address at any time.*
> - *The Admin is able to set the gas limit for passing messages via the Bridge to any value at any time.*
> - *The Bridge contract was not provided in the scope of this audit, so we are unable to provide an assessment of this contract with regards to security.*
> - *Additionally, relaying data between two EVM-based chains requires running off-chain logic in order to successfully deliver a message; We have not reviewed any off-chain logic in the scope of this audit, so we are unable to provide an assessment of the off-chain logic with regards to security.*
>
> *MediaEyeERC721Upgradeable and MediaEyeERC1155Upgradeable Contracts:*
>
> - *Anyone with or without a subscription is able to use these contracts to mint any amount of MeDIA eYe ERC721 or ERC1155 NFTs with custom data to any address; the recipient is declared the creator of the NFT.*
> - *NFT metadata that contains information about the NFT is stored using an off-chain URI endpoint.*

- *Any address with the Default Admin role or the Admin role is able to set the base URI value to any value at any time.*

CollectionFactory Contract:

- *Any user with a valid Subscription can use the CollectionFactory contract to deploy new MeDIA eYe NFT Collections in ERC721 or ERC1155 format at any time; the user is granted the Default Admin Role in the newly created Collection.*
- *The implementation code for the NFT Collections is set by the project team.*
- *Users are able to define an initial set of NFTs to mint when creating their Collection; users should exercise caution and limit the amount of NFTs included in the initial set to avoid hitting the block gas limit.*
- *Users with a Level Two Subscription are able to designate multiple addresses as Minters in their Collection.*
- *The project team is able to upgrade the implementation code used to create new NFT Collections at any time; changing the implementation code address only affects future deployments.*
- *The project team is able to set the contract address used to retrieve Subscription information at any time.*

> - *These contracts are intended to be used as the implementation code for the ERC721 and ERC1155 MeDIA eYe NFT Collections.*
> - *Any address with the Minter role is able to mint any amount of NFTs with custom data to any address; the recipient is declared the creator of the NFT.*
> - *NFT metadata that contains information about the NFT is stored using an off-chain URI endpoint.*
> - *Any address with the Default Admin role is able to set the base URI value to any value at any time.*
> - *The owner must have a Level Two Subscription in order to grant or revoke the Default Admin role or the Minter role to any user.*
>
> *MediaEyeCharities Contract:*
>
> - *This contract is used to maintain a list of addresses referred to as charities.*
> - *Any address with the Admin role can add or remove any address as a charity at anytime.*
> - *The address with the Default Admin role can grant or revoke the Admin role from any address at any time.*
>
> *MediaEyeMarketplaceInfo Contract:*
>
> - *The address with the Default Admin role can grant or revoke the Admin or the Setter role from any address at any time.*

- *Any address with the Setter role is able to set the royalty percentage up to the maximum acceptable value for any NFT that has not been sold yet.*
- *Any Setter is able to mark an NFT as sold at any time.*
- *Any Admin is able to add or remove any token as an accepted payment token at any time.*
- *Any Admin can set the maximum royalty percentage to any value up to 10% at any time.*

MediaEyeMarketplaceListings Contract:

- *Anyone can use this contract to list an NFT for sale at any time; users with a subscription are able to bundle multiple NFTs in a single listing.*
- *Users can specify various tokens for payment as long as they are approved payment methods.*
- *Any user can purchase the NFT with any of the specified tokens.*
- *A user can cancel any listings they have created at any time.*
- *Listed NFTs are held in the contract until they are sold or the listing is cancelled.*
- *Upon purchasing a listed NFT, a portion of the purchase amount is sent to the treasury wallet, a portion is sent to the creator as a royalty, a portion is sent to a charity address specified by the NFT owner, and another portion of the purchase value is sent to the secondary seller.*
- *The owner can set the treasury, MediaEyeFee, MediaEyeMarketplaceInfo, and MediaEyeCharity contract addresses to any address at any time.*

- *Any Admin is able to set the treasury fee percentage to any value up to 5% at any time.*

MediaEyeMarketplaceAuctions Contract:

- *Anyone can use this contract to auction an NFT at any time; users with a subscription are able to bundle multiple NFTs in a single auction.*
- *Each auction has a duration specified by the user.*
- *Users can specify various tokens for payment as long as they are approved payment methods. A buy-it-now price can also be specified in each payment token.*
- *Any user can purchase the NFT with any of the specified tokens.*
- *A user can cancel any auctions they have created at any time.*
- *Auctioned NFTs are held in the contract until they are sold or the auction is cancelled.*
- *Upon winning an auctioned NFT, a portion of the purchase amount is sent to the treasury wallet, a portion is sent to the creator as a royalty, a portion is sent to a charity address specified by the NFT owner, and another portion of the purchase value is sent to the secondary seller.*
- *The seller can use their private key as a signature to receive payment.*
- *The owner can set the treasury, MediaEyeFee, MediaEyeMarketplaceInfo, and MediaEyeCharity contract addresses to any address at any time.*
- *The owner can grant or revoke the Admin role from any address at any time.*

> *Airdrop:*
>
> - *The team intends to use the AirDropFactory contract to deploy ERC20, ERC721, or ERC1155 AirDrop contracts at any time.*
> - *The AirDropFactory contract allows anyone with the implementation code address to deploy their own AirDrop contract at any time; the user specifies the address that will serve as the Owner of the deployed contract, the implementation code address, the AirDrop data in the form of a MerkleTree, and whether or not the AirDrop is able to be cancelled.*
> - *The ERC20AirDrop contract is intended to be used as the implementation code for an ERC20 token AirDrop.*
> - *The ERC721AirDrop contract is intended to be used as the implementation code for an ERC721 token AirDrop.*
> - *The ERC1155AirDrop contract is intented to be used as the implementation code for an ERC1155 token AirDrop.*
> - *A user must supply a valid proof for the MerkleTree and the correct amount due to them in order to redeem their AirDrop from the contract; users may only redeem the full amount, and may only do so one time.*
> - *The owner of the deployed AirDrop implementation is able to change the data in the MerkleTree at any time.*

> - *If the AirDrop is able to be cancelled, the owner of the deployed AirDrop implementation is able to withdraw all the AirDrop tokens and ETH from the contract, and call selfdestruct() on the contract at any time.*
>
> *MediaEyeCanvas Contract:*
>
> - *Anyone can use this contract to purchase specific "Blocks" on a "canvas" using any payment token accepted by the platform; prices may vary based on the token used.*
> - *A portion of the funds is allocated to the charity fund, 10% of the funds are transferred to a fee wallet, and the remaining funds are transferred to a secondary fee wallet.*
> - *Any Admin is able to withdraw any ETH or ERC20 tokens from the contract and deliver them to any address at any time; these funds are intended to be transferred to various charity wallets.*
> - *Any Owner is able to set the address of the fee wallet at any time.*
> - *The secondary fee wallet address is able to set the address of the secondary fee wallet at any time.*
> - *Any Admin is able to add or remove any token as an accepted payment token at any time.*
> - *Any Admin is able to set the Block price for any accepted payment token to any value at any time.*
> - *Any Owner is able to grant or revoke any roles from any user at any time.*
> - *The project team should exercise caution when adding accepted payment tokens as to not use fee-on-transfer tokens or ERC777-compliant tokens; if fee-on-transfer tokens are used, the team must*

> *MediaEyeERC20 Contract:*
>
> - *The total supply of the MediaEyeERC20 token is initially minted to the owner upon deployment.*
> - *No minting or burn functions are present; the circulating supply can be reduced by sending tokens to the 0x..dead address, if desired.*
> - *The contract takes a fee on each transfer; whitelisted users are excluded from fees.*
> - *the owner is able to add or remove any address form the whitelist at any time.*
> - *The owner is able to set the tax rate to any value up to 10% at any time.*
>
> *EyeCohortFarming Contract:*
>
> - *This contract allows anyone to stake a single asset determined by the project team in exchange for rewards in various tokens.*
> - *On deployment, the owner must specify the staking token and one or many rewards tokens.*
> - *Each reward token has its own reward rate and its own end date. After a reward token's end date, rewards will no longer be distributed in that token.*
> - *Users are able to deposit and withdraw staking tokens at any time.*
> - *Users will receive a reward amount each second based on the amount staked; staking rewards can be calculated and transferred to the user at any time.*
> - *There must be an adequate amount of reward tokens in the contract in order to allow users to claim;*

> - *The owner is able to add a token as a reward token at any time.*
> - *The owner is able to set the reward rate for any reward token at any time.*
> - *The owner is able to set the end date and stop reward distribution for any reward token at any time.*
> - *The owner is able to withdraw the reward tokens from the contract at any time.*
> - *Appropriate use of ReentrancyGuard to prevent re-entrancy attacks in deposit and withdraw functions.*
> - *Excellent structuring of logic to allow for fee-on-transfer tokens to be used as the staking token.*
>
> *General Notes Across All Contracts:*
>
> - *The contracts utilize ReentrancyGuard to prevent against re-entrancy attacks.*
> - *The contracts comply with the relevant ERC721, ERC1155, and ERC20 token standards.*
> - *The team worked with us to implement changes related to gas optimization.*
> - *As the contracts are implemented with Solidity v0.8.x, they are protected from overflows/underflows.*

EXTERNAL THREAT RESULTS

Vulnerability Category	Notes	Result
Arbitrary Storage Write	N/A	PASS
Arbitrary Jump	N/A	PASS
Delegate Call to Untrusted Contract	N/A	PASS
Dependence on Predictable Variables	N/A	PASS
Deprecated Opcodes	N/A	PASS
Ether Thief	N/A	PASS
Exceptions	N/A	PASS
External Calls	N/A	PASS
Integer Over/Underflow	N/A	PASS

Vulnerability Category	Notes	Result
Multiple Sends	N/A	PASS
Suicide	N/A	PASS
State Change External Calls	N/A	PASS
Unchecked Retval	N/A	PASS
User Supplied Assertion	N/A	PASS
Critical Solidity Compiler	N/A	PASS
Overall Contract Safety		PASS

CONTRACT SOURCE SUMMARY AND VISUALIZATIONS

Name	Address/Source Code	Visualized (Hover-Zoom Recommended)
MediaEyeFee	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeHomeMediator	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeForeignMediator	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeXdaiMediator	GitHub (Not yet deployed	Function Graph.

MediaEyeERC721Upgradeable	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeERC1155Upgradeable	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeCollectionFactory	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeERC721CollectionUpgradeable	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.

	on mainnet)	Inheritance Chart.
MediaEyeMarketplaceInfo	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeCharities	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeMarketplaceListings	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeMarketplaceAuctions	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.

AirDropFactory	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
ERC20AirDrop	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
ERC721AirDrop	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
ERC1155AirDrop	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
MediaEyeCanvas	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.

MediaEyeERC20	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.
EyeCohortFarming	GitHub (Not yet deployed on mainnet)	Function Graph. Inheritance Chart.

GO HOME

Please note we are not associated with the Solidity programming language or the core team which develops the language.