Ethereal Audit Report

Version 1.0

Zkillua

Ethereal-NFTfi Audit Report

Zkillua

Feb 29th, 2024

Prepared by: [Zkillua]

Table of Contents

Table of Contents

Risk Classification

		Impact		
		High	Medium	Low
	High	H	H/M	M
Likelihood	Medium	H/M	M	M/L
	Low	M	M/L	L

Scope

- Code Base: https://github.com/owenThurm/Ethereal
- In Scope:

```
./src/
#-- ethereal.sol
```

Issues found

Severity	Number of issues found
High	2
Medium	2
Low	2
Info	3
Total	9

Findings

High

[H-1] Missing Pause and Unpause Functions.

Description:

- `Ethereal` contract inherits from OpenZeppelin's Pausable contract, which provides internal functions _pause() and _unpause() to toggle the contract's operational state. However, the inheriting contract does not expose any public or external functions to invoke these internal functions. As a result, there is no way for the contract owner or any other account to pause or unpause the contract, rendering the inherited Pausable functionality ineffective.

Impact:

- Without the ability to pause the contract, the owner cannot quickly respond to security threats, bugs, or exploits that may arise .This increases the risk of loss or damage in the event of an emergency.

Proof of Concept:

- The Ethereal.sol contract code does not contain any functions that call _pause() or _unpause(). A review of the contract's functions reveals that none allow changing the contract's paused state:

Recommended mitigation:

- Implement pause and unpause functions in the contract.

```
++ function pause() public onlyOwner {
     _pause();
}

++ function unpause() public onlyOwner {
     _unpause();
}
```

[H-2]: `Ethereal::fees` should be seperately calculated for `_redeemEth` and `_redeemWstEth` function transactions, instead of using single variable.

Description:

- For native eth transactions `metadata[_tokenId].balance` = msg.value, where as for WstEth token transactions ,`metadata[_tokenId].balance`= IwstETH(wstETH).balanceOf(address(this)) - preBalance. Both are different assets and hold different values. The single fee variable is used in calculating the accumulated fees from both redeemEth and redeemWstEth transactions.

Impact:

- Since Fee variable calculates fees of both transactions together, whenever owner withdraws the fees using `withdrawFees` function, payout address withdraws total fees from contract's Eth Balance instead of withdrawing from both native Eth and WstEth tokens, hence resulting in draining of Eth balance of contract.

Proof of Code: - Assume 3 users: user1,user2,user3

1. create 2 gems 1 backed by ether, another by wstEth, both with same denomination=10*1e18,and redeem fee=1000.
2. user1 mints 1 gem backed by eth, user2 and 3 mint 1 gem each backed by wstEth. –> contract's eth balance=10,
3. user2,user3 shall redeem their gems causing fees=2 –> contract's eth balance=10,
4. owners calls for withdraw of accumulated fees => draws 2 "Eth" from contract's balance. –> contract's Eth balance=10-2=8 Eth

5. user1 calls for redeem to claim 9 Eth but transaction reverts since contract's balance is only 8 Eth.

Recommended Mitigations:

- Should calculate fees seperately,for example use 2 variables , one to track fees accumulated in _redeemEth and another to track fees accumulated in _redeemWstEth.
- `IwstETH(wstETH).transfer(payout, wstETHFees)` should be used for paying fee in wsteth, where wstETHFees is fees accumulated in _redeemWstEth transactions

Medium

[M-1] Centralization Risk Due to Owner Privileges (Single Point of Failure)

Description:

- The `Ethereal` contract utilizes the `onlyOwner` modifier for several critical administrative functions, such as `createCollection`, `createGem`, `updateCollection`, `updateGem`, `ceaseGem`, `setWstEth`, `setPayout`, `withdrawFees`, and `approveWstEth`.

Impact:

- If the owner's private key is compromised, an attacker could potentially take over the contract, manipulate its settings, drain funds, or disrupt operations.
- Owner's with malicious Intent may even attempt a rugpull by changing contract addresses.

Recommended Mitigation:

- Protocols should consider implementing mechanisms like Multi-signature control, DAOs,Timelocks etc. This increases trust among users, realising true decentralisation nature of blockchain.

[M-2] Insufficient Validations in updateCollection and updateGem Function can cause Unintended consequences to the state of protocol.

Description:

- `functions::updateCollection` & `updateGem` are used to update properties of gem whenever the owner intends to. Inconsistencies may arise when owner updates gem's properties.

1. Ceased Gem can be reactivated using updategem function by setting `gems[_id].active=true`.This was unintended as ceaseGem function intends to make gems no longer mintable.
2. The `collections[_id].ethereum` can also be updated without restrictions. If the backing of the gem is changed from native eth to wsteth after users mints, it creates state inconsistency of the contract.

Recommended Mitigation:

- Certain properties of gems and collections should be kept immutable, or sufficient checks should be added allowing the owner to change properties only prior to minting.

Low

[L-1] Solidity pragma should be specific, not wide

Description:

- Consider using a specific version of Solidity in contracts instead of a wide version. For example, instead of pragma solidity ^0.8.0;, use pragma solidity 0.8.0;

[L-2] Setter-functions must emit events.

Description:

- Setter function Ethereal.setWstEth(address) (src/ethereal.sol#L249-252) does not emit an event.
- Ethereal.setPayout(address) (src/ethereal.sol#254-257) does not emit an event.

Informational

[I-1]Functions not used internally could be marked external

Description: - Found in src/ethereal.sol Line: 214 - Found in src/ethereal.sol Line: 218 - Found in src/ethereal.sol Line: 226 Line: 230

[I-2]: Constants should be defined and used instead of literals

- Found in src/ethereal.sol Line: 184 Line: 195 Line: 273

[I-3]: Event is missing `indexed` fields

Index event fields make the field more quickly accessible to off-chain tools that parse events.

- Found in src/ethereal.sol Line: 19

```
1       event Transfer(address indexed from, address indexed to,
           uint256 tokens);
```

- Found in src/ethereal.sol Line: 20

```
1       event Approval(address indexed tokenOwner, address indexed
           spender, uint256 tokens);
```