Cumulative Merkle drop audit

1 Sep 2021, Igor Gulamov

Introduction

Igor Gulamov conducted the audit of 1inch Cumulative Merkle drop smart contracts.

This review was performed by an independent reviewer under fixed rate.

Scope

Smart contracts from 1026fe507d829ac1708a3393f85f7b91620b0ed8

Issues

Major

CumulativeMerkleDrop128.sol#L38

2nd preimage resistance for this tree is 128 bits, but collision resistance is only 64 bits. In the case if 3rd parties can maniputate leaves to set any special arbitrary value, usage of this tree is not safe. Here is the attack scheme:

1. The attacker find an event to trigger fixed amount 1 token airdrop to any address.
2. The attaker find collision, for example $\text{hash}(\text{address}_1, 1\ \text{token})=\text{hash}(\text{address}_2, 1e6\ \text{token})$
3. The attacker trigger 1 token airdrop to address_1 and withdraw 1e6 token to address_2.

To find this collision the attacker should bruteforce about 2^{64} hashes.

We propose adding salt to leaves of the Merkle tree, replacing ```solidity=38 bytes16 leaf = _keccak128(abi.encodePacked(account, cumulativeAmount))

~~~
 with
```solidity=38
bytes16 leaf = _keccak128(abi.encodePacked(account, cumulativeAmount, salt))
~~~

`salt` should be randomly generated by airdrop operator, and it should be unpredictable by all other parties.

Comment

CumulativeMerkleDrop128.sol#L54-L71

CumulativeMerkleDrop160.sol#L54-L71

We propose removing commented source code.

Severity Terms

Comment

Comment issues are generally subjective in nature, or potentially deal with topics like "best practices" or "readability". Comment issues in general will not indicate an actual problem or bug in code.

The maintainers should use their own judgment as to whether addressing these issues improves the codebase.

Warning

Warning issues are generally objective in nature but do not represent actual bugs or security problems.

These issues should be addressed unless there is a clear reason not to.

Major

Major issues will be things like bugs or security vulnerabilities. These issues may not be directly exploitable, or may require a certain condition to arise in order to be exploited.

Left unaddressed these issues are highly likely to cause problems with the operation of the contract or lead to a situation which allows the system to be exploited in some way.

Critical

Critical issues are directly exploitable bugs or security vulnerabilities.

Left unaddressed these issues are highly likely or guaranteed to cause major problems or potentially a full failure in the operations of the contract.