Crypton (&Studio)

THENA - Smart Contract Audit

Prepared by:
CryptonStudio

Date of Engagement: 28.10.2022

Contents

Executive overview

The security assessment was scoped to the smart contract of THENA protocol. At the time of the audit, all source files were located at the **link**.

The team at CryptonStudio was provided a one week timeframe for the engagement and assigned three full time security engineers to audit the security of the smart contract. The security engineers are blockchain and smart contract security experts, with experience in advanced penetration testing, smart contract hacking, and have a deep knowledge in multiple blockchain protocols.

The purpose of this audit to achieve the following:

- Ensure that smart contract functions are intended.

- Identify potential security issues with the smart contracts.

In summary, CryptonStudio identified few security risks, and recommends performing further testing to validate extended safety and correctness in context to the whole set of contracts.

Vulnerabilities or issues observed by CryptonStudio are ranked based on the risk assessment methodology by measuring the likelihood of a security incident, and the impact should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

CRITICAL	**HIGH**	**MEDIUM**	**LOW**	**INFORMATIONAL**

Scope

The security assessment was scoped to the smart contract:

- ExternalBribe.sol
- Gauge.sol
- InternalBribe.sol
- MasterChef.sol
- Minter.sol
- Pair.sol
- PairFees.sol
- RewardsDistributor.sol
- Router.sol
- StakingNFTFeeConverter.sol
- Thena.sol
- ThenaGovernor.sol
- ThenaLibrary.sol
- VeArtProxy.sol
- Voter.sol
- VotingEscrow.sol
- WrappedExternalBribe.sol
- BribeFactory.sol
- GaugeFactory.sol
- PairFactory.sol
- WrappedExternalBribeFactory.sol
- L2Governor.sol
- L2GovernorCountingSimple.sol
- L2GovernorVotes.sol
- L2GovernorVotesQuorumFraction.sol
- MerkleClaim.sol
- RedemptionReceiver.sol
- RedemptionSender.sol

Assessment summary & findings overview

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	1	1	2

Findings & tech details

MSCH-01

Smart conract — MasterChef.sol

Severity — Informational

Description

`SafeMath` is no longer needed starting with Solidity 0.8. The compiler now has built-in overflow checking.

TLIB-01

Smart conract — ThenaLibrary.sol

Severity — Low

Description

_f(x0, y) and _d(x0, y) functions may lose precision.

Recommendation

_f(x0, y) and _d(x0, y) functions may lose precision.
Recommendation: Rewrite _f(x0, y) and _d(x0, y) functions.

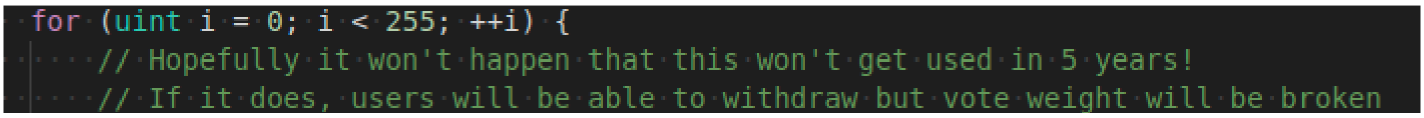
VESC-01

Smart conract — VotingEscrow.sol

Severity — Medium

Description

Function _checkpoint has vulnerability in the loop.



```
for (uint i = 0; i < 255; ++i) {
    // Hopefully it won't happen that this won't get used in 5 years!
    // If it does, users will be able to withdraw but vote weight will be broken
```

Recommendation

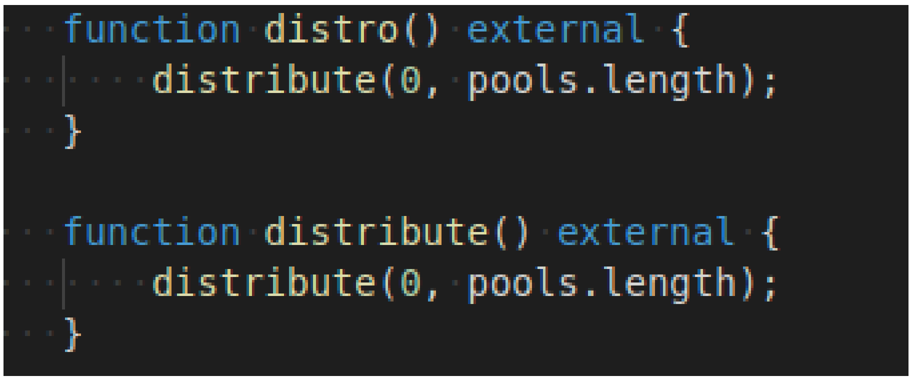
Rewrite this function.

VTR-01

Smart conract — Voter.sol

Description

distro() and distribute() functions are exactly the same.



```
  function distro() external {
      distribute(0, pools.length);
  }

  function distribute() external {
      distribute(0, pools.length);
  }
```

Recommendation

Remove one of the functions.