SECURITY AUDIT OF

HEROESTD SMART CONTRACT

Public Report

Nov 01, 2021

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
Ether (ETH)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
ERC20	ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain.

EXECUTIVE SUMMARY

This Security Audit Report prepared by Verichains Lab on Nov 01, 2021. We would like to thank the Heroes TD for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the HeroesTD Smart Contract. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified no vulnerable issues in the smart contracts code.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About HeroesTD Smart Contract

Heroes TD is a collectible tower defense strategy game, with a brand new play style. You can collect and build your deck to win. You can both enjoy the game and earn money from it as well. In Heroes TD, players can join a community to play with each other. Players can also create new unique NFT content and trade them with other players to get real money, and most importantly, have a lot of fun!

HeroesTD is an ERC20 token that Heroes TD players can use to buy heroes, upgrade them and earn rewards.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the HeroesTD Smart Contract.

The audited contract is the HeroesTD Smart Contract that deployed on Binance Smart Chain Mainnet at address 0x5E2689412Fae5c29BD575fbe1d5C1CD1e0622A8f. The details of the deployed smart contract are listed in Table 1.

FIELD	VALUE
Contract Name	HeroesTD
Contract Address	0x5E2689412Fae5c29BD575fbe1d5C1CD1e0622A8f
Compiler Version	v0.8.2+commit.661d1103
Optimization Enabled	Yes with 200 runs
Explorer	https://bscscan.com/address/0x5E2689412Fae5c29BD575fbe1d5C1CD1e0622A8f

Table 1. The deployed smart contract details

1.3. Audit methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 2. Severity levels

1.4. Disclaimer

Please note that security auditing cannot uncover all existing vulnerabilities, and even an audit in which no vulnerabilities are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

2. AUDIT RESULT

2.1. Overview

Table 2 lists some properties of the audited HeroesTD Smart Contract (as of the report writing time).

PROPERTY	VALUE
Name	HeroesTD
Symbol	HTD
Decimals	18
Total Supply	468,000,000 (x10^{18}) Note: the number of decimals is 18, so the total representation token will be 468,000,000 or 468 million.

Table 3. The HeroesTD Smart Contract properties

2.2. Contract codes

The HeroesTD Smart Contract was written in Solidity language, with the required version to be 0.8.2.

2.3. Findings

During the audit process, the audit team found no vulnerability in the given version of HeroesTD Smart Contract.

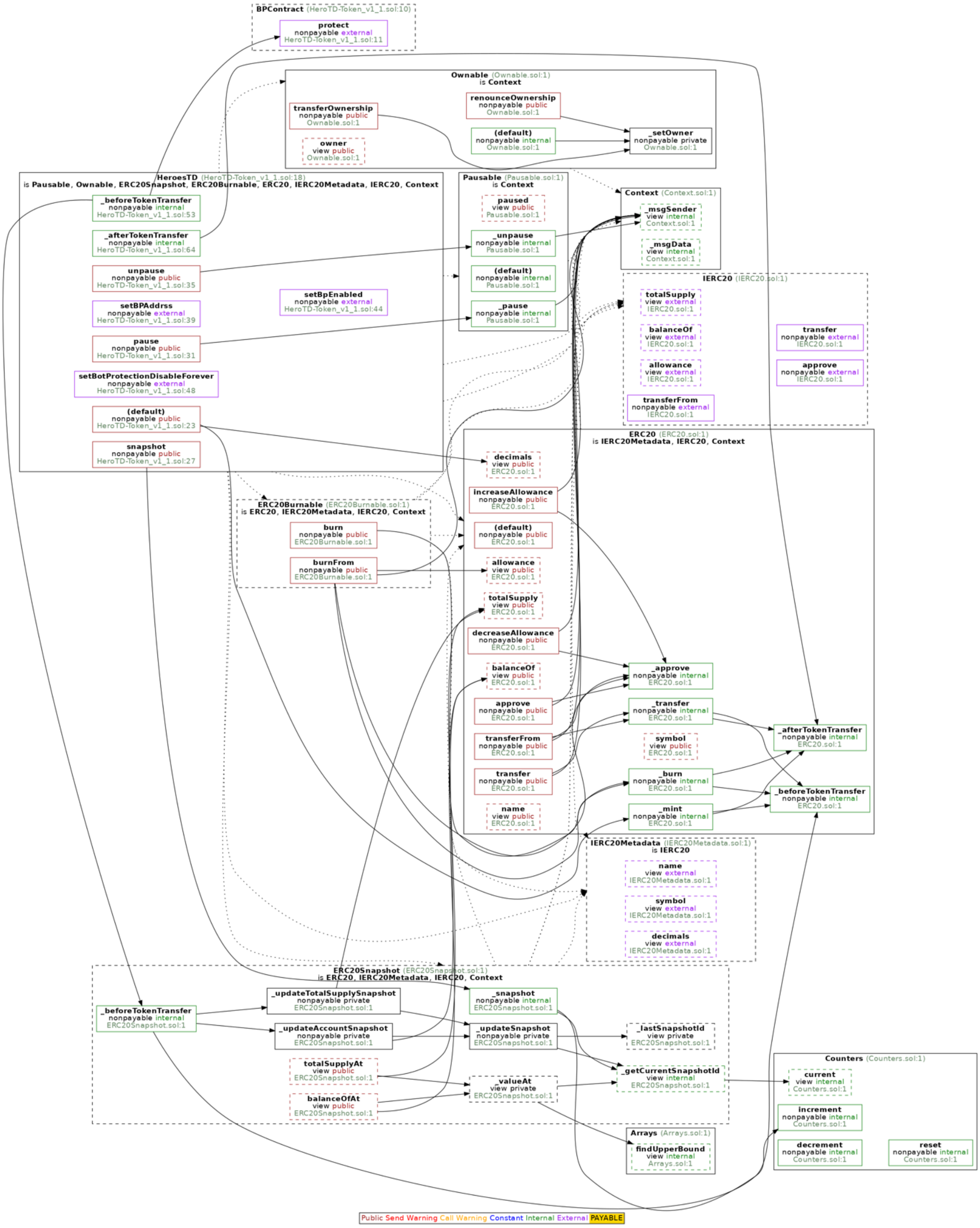
2.4. Additional notes and recommendations

2.4.1. BPContract function INFORMATIVE

Since we do not control the logic of the BPContract, there is no guarantee that BPContract will not contain any security related issues. With the current context, in case the BPContract is compromised, there is not yet a way to exploit the HeroesTD Smart Contract, but we still note that here as a warning for avoiding any related issue in the future.

By the way, if having any issue, the BPContract function can be disabled easily anytime by the contract owner using the setBpEnabled function. In addition, BPContract is only used in a short time in token public sale IDO then the contract owner will disable it forever by the setBotProtectionDisableForever function.

APPENDIX



BPContract (HeroTD-Token_v1_1.sol:10)
protect nonpayable external HeroTD-Token_v1_1.sol:11
Ownable (Ownable.sol:1) is Context
transferOwnership nonpayable public Ownable.sol:1
renounceOwnership nonpayable public Ownable.sol:1
(default) nonpayable internal Ownable.sol:1
_setOwner nonpayable private Ownable.sol:1
owner view public Ownable.sol:1
HeroesTD (HeroTD-Token_v1_1.sol:18) is Pausable, Ownable, ERC20Snapshot, ERC20Burnable, ERC20, IERC20Metadata, IERC20, Context
_beforeTokenTransfer nonpayable internal HeroTD-Token_v1_1.sol:53
_afterTokenTransfer nonpayable internal HeroTD-Token_v1_1.sol:64
unpause nonpayable public HeroTD-Token_v1_1.sol:35
setBPAddrss nonpayable external HeroTD-Token_v1_1.sol:39
setBpEnabled nonpayable external HeroTD-Token_v1_1.sol:44
pause nonpayable public HeroTD-Token_v1_1.sol:31
setBotProtectionDisableForever nonpayable external HeroTD-Token_v1_1.sol:48
(default) nonpayable public HeroTD-Token_v1_1.sol:23
snapshot nonpayable public HeroTD-Token_v1_1.sol:27
Pausable (Pausable.sol:1) is Context
paused view public Pausable.sol:1
_unpause nonpayable internal Pausable.sol:1
(default) nonpayable internal Pausable.sol:1
_pause nonpayable internal Pausable.sol:1
Context (Context.sol:1)
_msgSender view internal Context.sol:1
_msgData view internal Context.sol:1
IERC20 (IERC20.sol:1)
totalSupply view external IERC20.sol:1
balanceOf view external IERC20.sol:1
allowance view external IERC20.sol:1
transferFrom nonpayable external IERC20.sol:1
transfer nonpayable external IERC20.sol:1
approve nonpayable external IERC20.sol:1
ERC20 (ERC20.sol:1) is IERC20Metadata, IERC20, Context
decimals view public ERC20.sol:1
increaseAllowance nonpayable public ERC20.sol:1
(default) nonpayable public ERC20.sol:1
allowance view public ERC20.sol:1
totalSupply view public ERC20.sol:1
decreaseAllowance nonpayable public ERC20.sol:1
balanceOf view public ERC20.sol:1
approve nonpayable public ERC20.sol:1
transferFrom nonpayable public ERC20.sol:1
transfer nonpayable public ERC20.sol:1
name view public ERC20.sol:1
_approve nonpayable internal ERC20.sol:1
_transfer nonpayable internal ERC20.sol:1
symbol view public ERC20.sol:1
_burn nonpayable internal ERC20.sol:1
_mint nonpayable internal ERC20.sol:1
_afterTokenTransfer nonpayable internal ERC20.sol:1
_beforeTokenTransfer nonpayable internal ERC20.sol:1
ERC20Burnable (ERC20Burnable.sol:1) is ERC20, IERC20Metadata, IERC20, Context
burn nonpayable public ERC20Burnable.sol:1
burnFrom nonpayable public ERC20Burnable.sol:1
IERC20Metadata (IERC20Metadata.sol:1) is IERC20
name view external IERC20Metadata.sol:1
symbol view external IERC20Metadata.sol:1
decimals view external IERC20Metadata.sol:1
ERC20Snapshot (ERC20Snapshot.sol:1) is ERC20, IERC20Metadata, IERC20, Context
_beforeTokenTransfer nonpayable internal ERC20Snapshot.sol:1
_updateTotalSupplySnapshot nonpayable private ERC20Snapshot.sol:1
_updateAccountSnapshot nonpayable private ERC20Snapshot.sol:1
totalSupplyAt view public ERC20Snapshot.sol:1
balanceOfAt view public ERC20Snapshot.sol:1
_snapshot nonpayable internal ERC20Snapshot.sol:1
_updateSnapshot nonpayable private ERC20Snapshot.sol:1
_valueAt view private ERC20Snapshot.sol:1
_lastSnapshotId view private ERC20Snapshot.sol:1
_getCurrentSnapshotId view internal ERC20Snapshot.sol:1
Arrays (Arrays.sol:1)
findUpperBound view internal Arrays.sol:1
Counters (Counters.sol:1)
current view internal Counters.sol:1
increment nonpayable internal Counters.sol:1
decrement nonpayable internal Counters.sol:1
reset nonpayable internal Counters.sol:1
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 1. HeroesTD Smart Contract call graph

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Oct 28, 2021*	Public Report	Verichains Lab
1.1	*Nov 01, 2021*	Public Report	Verichains Lab

Table 4. Report versions history