HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Xsigma
Date: February 9th, 2021

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for xSigma.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Token, Governance, TimeLock, Defi, Proxy
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/xSigmaLabs/xsigma-dex-contracts
Commit	FAAB1E2060FEB01E69C34960E1676A6D758E9763
Deployed contract	
Timeline	05 FEB 2021 - 09 FEB 2021
Changelog	09 FEB 2021 - INITIAL AUDIT 15 FEB 2021 - REMEDIATION CHECK

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Introduction....4
Scope....4
Executive Summary....5
Severity Definitions....6
AS-IS overview....7
Conclusion....32
Disclaimers....33

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by Xsigma (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of Customer's smart contract and its code review conducted between February 5th, 2021 - February 9th, 2021.

Scope

The scope of the project is smart contracts in the repository:

```
Contract deployment address:
Repository
File:
      DAO/Governor.sol
      DAO/TimeLock.sol
      SigMasterChef.sol
      SigThreePoolProxy.sol
      SigToken.sol
```

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ DoS with (Unexpected) Throw ▪ DoS with Block Gas Limit ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ Costly Loop ▪ ERC20 API violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency ▪ Data Consistency

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Functional review	■ Business Logics Review ■ Functionality Checks ■ Access Control & Authorization ■ Escrow manipulation ■ Token Supply manipulation ■ Assets integrity ■ User Balances manipulation ■ Kill-Switch Mechanism ■ Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contracts are well-secured.

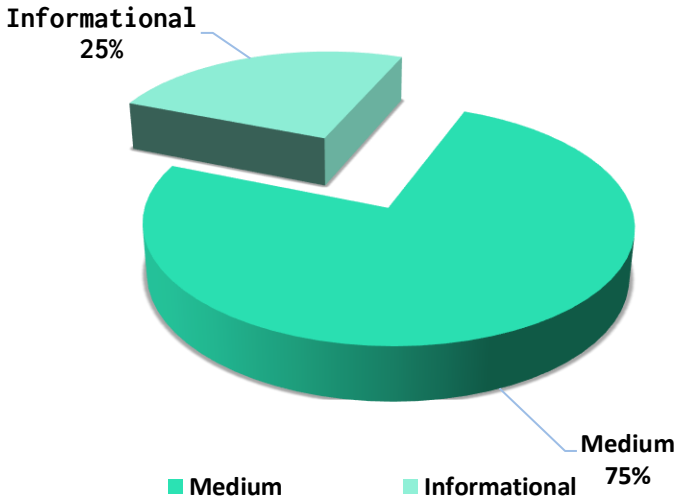
Insecure	Poor secured	Secured	Well-secured

You are

Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. A general overview is presented in AS-IS section, and all found issues can be found in the Audit overview section.

Security engineers found **2** medium, **1** informational issue during the audit.

Graph 1. The distribution of vulnerabilities after the first review.



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations, and info statements can't affect smart contract execution and can be ignored.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

AS-IS overview

Timelock.sol

Description

Timelock queues and executes transactions.

Imports

Timelock has following imports:

- SafeMath.sol - from the OpenZeppelin.

Inheritance

Timelock does not inherit anything.

Usages

Timelock contract has following usages:

- SafeMath for uint.

Structs

Timelock contract has no data structures.

Enums

Timelock contract has no enums.

Events

Timelock contract has following events:

- event NewAdmin(address indexed newAdmin);
- event NewPendingAdmin(address indexed newPendingAdmin);
- event NewDelay(uint indexed newDelay);
- event CancelTransaction(bytes32 indexed txHash, address indexed target, uint value, string signature, bytes data, uint eta);
- event ExecuteTransaction(bytes32 indexed txHash, address indexed target, uint value, string signature, bytes data, uint eta);
- event QueueTransaction(bytes32 indexed txHash, address indexed target, uint value, string signature, bytes data, uint eta);

Modifiers

Timelock has no modifiers.

Fields

Timelock contract has following fields and constants:

- uint public constant GRACE_PERIOD = 14 days;
- uint public constant MINIMUM_DELAY = 2 days;
- uint public constant MAXIMUM_DELAY = 30 days;
- address public admin - an admin address.
- address public pendingAdmin - a pending admin.
- uint public delay - delay between a transaction queueing and execution.
- mapping (bytes32 => bool) public queuedTransactions - queued transactions.

Functions
Timelock has following public functions:

- ***constructor***
  Description
  Inits the contract and sets default parameters.
  Visibility
  public
  Input parameters
  -
  - uint delay_ - delay between a transaction queuing and execution.

  Constraints
  - A `delay_` value should be between DELAY and MAXIMUM_DELAY.

  Events emit
  None
  Output
  None
- ***receive***
  Description
  Forbids ETH transfers.
- ***setDelay***
  Description
  Sets a delay.
  Visibility
  public
  Input parameters
  - uint delay_ - delay between a transaction queuing and execution.

  Constraints

- A message sender should be the contract itself.
- A `delay_` value should be between DELAY and MAXIMUM_DELAY.

Events emit
Emits the `NewDelay` event.
Output
None

- ***acceptAdmin***
  Description
  Accept the admin permissions.
  Visibility
  public
  Input parameters
  None
  Constraints
  - A message sender should be a pending admin.

  Events emit
  Emits the `NewAdmin` event.
  Output
  None
- ***setPendingAdmin***
  Description
  Accept the admin permissions.
  Visibility
  public
  Input parameters
  - address pendingAdmin_ - a pending admin address.

  Constraints
  - A message sender should be the contract itself.

  Events emit
  Emits the `NewPendingAdmin` event.
  Output
  None
- ***queueTransaction***
  Description
  Add a new transaction to the queue.
  Visibility
  public
  Input parameters
  - address target - a tx target.
  - uint value - a tx value.
  - string memory signature - a method signature.
  - bytes memory data - a tx data.
  - uint eta - a minimum delay between a tx queuing and execution.

  Constraints

- A message sender should be admin.
- `eta` should be more than current time plus delay value.

Events emit
Emits the `QueueTransaction` event.
Output
bytes32 - a tx hash.

- ***cancelTransaction***
  Description
  Cancel a transaction.
  Visibility
  public
  Input parameters
  - address target - a tx target.
  - uint value - a tx value.
  - string memory signature - a method signature.
  - bytes memory data - a tx data.
  - uint eta - a minimum delay between a tx queuing and execution.

  Constraints
  - A message sender should be admin.

  Events emit
  Emits the `CancelTransaction` event.
  Output
  None
- ***executeTransaction***
  Description
  Execute a transaction.
  Visibility
  public
  Input parameters
  - address target - a tx target.
  - uint value - a tx value.
  - string memory signature - a method signature.
  - bytes memory data - a tx data.
  - uint eta - a minimum delay between a tx queuing and execution.

  Constraints
  - A message sender should be admin.
  - A transaction should be queued.
  - Current timestamp should be between `eta` and `eta` + GRACE_PERIOD.

  Events emit
  Emits the `ExecuteTransaction` event.
  Output
  None

Governor.sol

Description

Governor allows to vote for specific actions using KeyFi token votes.

Imports

Governor has no imports.

Inheritance

Governor does not inherit anything.

Usages

Governor contract has no usages.

Structs

Governor contract has following data structures:

- Proposal
- Receipt

Enums

Governor contract has following enums:

- ProposalState

Events

Governor contract has following events:

- event ProposalCreated(uint id, address proposer, address[] targets, uint[] values, string[] signatures, bytes[] calldatas, uint startBlock, uint endBlock, string description);
- event VoteCast(address voter, uint proposalId, bool support, uint votes);
- event ProposalCanceled(uint id);
- event ProposalQueued(uint id, uint eta);
- event ProposalExecuted(uint id);

Modifiers

Governor has no modifiers.

Fields

Governor contract has following fields and constants:

- string public constant name = "XSigma Governor " - name of the contract
- TimelockInterface public timelock - the timelock address.
- TokenInterface public keyfi - the KeyFi token address.
- address public guardian - a guardian address.
- uint public proposalCount - the total number of proposals
- mapping (uint => Proposal) public proposals - all proposals.
- mapping (address => uint) public latestProposalIds - proposals of an address.
- bytes32 public constant DOMAIN_TYPEHASH = keccak256("EIP712Domain(string name,uint256 chainId,address verifyingContract)");
- bytes32 public constant BALLOT_TYPEHASH = keccak256("Ballot(uint256 proposalId,bool support)");

Functions
Governor has following public functions:

- ***constructor***
  Description
  Inits the contract and sets default parameters.
  Visibility
  public
  Input parameters
  - address timelock_ - the time contract address.
  - address keyfi_ - the KeyFi token address.
  - address guardian_ - a guardian address.

  Constraints
  None
  Events emit
  None
  Output
  None
- ***propose***
  Description
  Propose a new vote.
  Visibility
  public
  Input parameters
  - address[] memory targets - transaction targets.
  - uint[] memory values - transaction values.
  - string[] memory signatures - functions to be executed.
  - bytes[] memory calldatas - calldata of calls.
  - string memory description - a propose description.

Constraints
- A proposer votes should be above the proposal threshold.
- `targets`, `values`, `signatures`, `calldatas` should be of the same length.
- `targets` length should be more than 0.
- `targets` length should be less than proposalMaxOperations.
- A proposer can have one live.

Events emit
Emits the `ProposalCreated` event.
Output
- uint - a propose id.

- ***queue***
Description
Queue a proposal for the execution.
Visibility
public
Input parameters
- uint proposalId - a proposal id.

Constraints
- A proposal can only be queued only if it is succeeded.
- A proposal should not be queued yet.

Events emit
Emits the `ProposalQueued` event.
Output
None

- ***execute***
Description
Execute a proposal.
Visibility
public
Input parameters
- uint proposalId - a proposal id.

Constraints
- A proposal should be queued.
- A proposal should not be executed yet.

Events emit
Emits the `ProposalExecuted` event.
Output
None

- ***cancel***
Description
Cancel a proposal.
Visibility
public
Input parameters

- uint proposalId - a proposal id.

Constraints

- A proposal should not be executed yet.
- A message sender should be a guardian or a proposer votes become less than threshold.

Events emit

Emits the `ProposalCanceled` event.

Output

None

- ***getActions***

Description

Get actions of a specified proposal.

Visibility

public view

Input parameters

- uint proposalId - a proposal id.

Constraints

None

Events emit

None

Output

- address[] memory targets
- uint[] memory values
- string[] memory signatures
- bytes[] memory calldatas

- ***getReceipt***

Description

Get a vote results of a `voter` in a provided proposal.

Visibility

public view

Input parameters

- uint proposalId - a proposal id.
- address voter - a voter address.

Constraints

None

Events emit

None

Output

- Receipt memory

- ***state***

Description

Get a state of a given proposal.

Visibility

public view

Input parameters

- uint proposalId - a proposal id.

Constraints
None
Events emit
None
Output
  - ProposalState

- ***state***
Description
Get a state of a given proposal.
Visibility
public view
Input parameters
  - uint proposalId - a proposal id.
Constraints
None
Events emit
None
Output
  - ProposalState
- ***castVote***
Description
Participate in a vote.
Visibility
public
Input parameters
  - uint proposalId - a proposal id.
  - bool support
Constraints
  - A proposal should be active.
  - A voter should not participate yet.
Events emit
Emits the `VoteCast` event
Output
None
- ***castVoteBySig***
Description
Participate in a vote on behalf of another voter.
Visibility
public
Input parameters
  - uint proposalId - a proposal id.
  - bool support
  - uint8 v - part of a signature.
  - bytes32 r - part of a signature.
  - bytes32 s - part of a signature.
Constraints

- A proposal should be active.
- A voter should not participate yet.

Events emit
Emits the `VoteCast` event
Output
None

- ***__acceptAdmin***
  Description
  Accept admin rights of the Timelock contract.
  Visibility
  public
  Input parameters
  None
  Constraints
  - A message sender should be a guardian.

  Events emit
  None
  Output
  None
- ***__abdicate***
  Description
  Removes a guardian address.
  Visibility
  public
  Input parameters
  None
  Constraints
  - A message sender should be a guardian.

  Events emit
  None
  Output
  None
- ***__queueSetTimelockPendingAdmin***
  Description
  Queue a new pending admin of the Timelock contract.
  Visibility
  public
  Input parameters
  - address newPendingAdmin - an address of the new admin.
  - uint eta - execution time.

  Constraints
  - A message sender should be a guardian.

  Events emit
  None
  Output
  None

- ***executeSetTimelockPendingAdmin***
  Description
  Set a new pending admin of the Timelock contract.
  Visibility
  public
  Input parameters
  - address newPendingAdmin - an address of the new admin.
  - uint eta - execution time.

  Constraints
  - A message sender should be a guardian.
  - A transaction should be previously queued.

  Events emit
  None
  Output
  None

SigMasterChef.sol

Description

SigMasterChef is a liquidity pool with rewards in Sig token.

Imports

SigMasterChef has following imports:

- @openzeppelin/contracts/token/ERC20/IERC20.sol
- @openzeppelin/contracts/token/ERC20/SafeERC20.sol
- @openzeppelin/contracts/utils/EnumerableSet.sol
- @openzeppelin/contracts/math/SafeMath.sol
- @openzeppelin/contracts/access/Ownable.sol
- ./SigToken.sol

Inheritance

SigMasterChef is Ownable.

Usages

SigMasterChef contract has following usages:

- SafeMath for uint256
- SafeERC20 for IERC20

Structs

SigMasterChef contract has following data structures:

- UserInfo

- PoolInfo

Enums

SigMasterChef contract has no enums.

Events

SigMasterChef contract has following events:

- Deposit
- Withdraw
- EmergencyWithdraw

Modifiers

SigMasterChef has no custom modifiers.

Fields

SigMasterChef contract has following fields and constants:

- SigToken public sushi
- address public devaddr
- address public sharedVault
- PoolInfo[] public poolInfo
- mapping (uint256 => mapping (address => UserInfo)) public userInfo
- uint256 public totalAllocPoint = 0
- uint256 public startBlock
- uint256 public rewardTimeFactor = 1
- uint256 constant MIN_REWARD_TIME_FACTOR = 1
- uint256 constant MAX_REWARD_TIME_FACTOR = 40

Functions
SigMasterChef has following public functions:

- ***constructor***
  Description
  Sets initial values of the contract.
  Visibility
  public
  Input parameters
  - SigToken _sushi,
  - address _devaddr,
  - address _sharedVault
  - uint256 _startBlock

  Constraints

None
Events emit
None
Output
None

- ***poolLength***
  Description
  Returns a number of pools.
  Visibility
  external view
  Input parameters
  None
  Constraints
  None
  Events emit
  None
  Output
  - uint256 - a number of pools.
- ***changeFactor***
  Description
  Updates the *rewardTimeFactor*.
  Visibility
  public
  Input parameters
  None
  Constraints
  - onlyOwner modifier.

  Events emit
  None
  Output
  None
- ***add***
  Description
  Add a new lp to the pool.
  Visibility
  public
  Input parameters
  - uint256 _allocPoint
  - IERC20 _lpToken
  - bool _withUpdate

  Constraints
  - onlyOwner modifier.

  Events emit
  None
  Output
  None

- ***set***
  Description
  Update the given pool's allocation point
  Visibility
  public
  Input parameters
  - uint256 _pid
  - uint256 _allocPoint
  - bool _withUpdate

  Constraints
  - onlyOwner modifier.

  Events emit
  None
  Output
  None
- ***pendingSushi***
  Description
  Returns pending tokens of a _user for a _pid reward pool.
  Visibility
  external view
  Input parameters
  - uint256 _pid
  - address _user

  Constraints
  None
  Events emit
  None
  Output
  - uint256 - available tokens.
- ***pendingSushi***
  Description
  Returns pending tokens of a _user for a _pid reward pool for a specified block number.
  Visibility
  external view
  Input parameters
  - uint256 _pid
  - address _user
  - uint256 blockNumber

  Constraints
  - blockNumber should exceed current block.

  Events emit
  None
  Output
  - uint256 - available tokens.
- ***massUpdatePools***

Description
Update reward variables for all pools.
Visibility
public
Input parameters
None
Constraints
None
Events emit
None
Output
None

- ***updatePool***
  Description
  Update reward variables of the given pool to be up-to-date.
  Visibility
  public
  Input parameters
    - uint256 _pid

  Constraints
  None
  Events emit
  None
  Output
  None
- ***deposit***
  Description
  Deposit LP tokens.
  Visibility
  public
  Input parameters
    - uint256 _pid
    - uint256 _amount

  Constraints
  None
  Events emit
  Emits the Deposit event.
  Output
  None
- ***withdraw***
  Description
  Withdraw LP tokens.
  Visibility
  public
  Input parameters
    - uint256 _pid

    - uint256 _amount

  Constraints
    - An _amount should not exceed a user balance of a _pid pool

  Events emit
  Emits the Withdraw event.
  Output
  None

- ***emergencyWithdraw***
  Description
  Withdraw LP tokens without a reward.
  Visibility
  public
  Input parameters
    - uint256 _pid

  Constraints
  None
  Events emit
  Emits the EmergencyWithdraw event.
  Output
  None

SigToken.sol

Description

SigToken.sol contains a Sig token contact.

SigToken is a token with following parameters:

- Name: xSigma
- Symbol: SIG
- Decimals: 18

The SigToken has voting functionality.
The SigToken mints tokens available for the team on the contract creation. The sum is **not limited**.

Imports

SigToken contract has following imports:

- @openzeppelin/contracts/token/ERC20/ERC20.sol
- @openzeppelin/contracts/token/ERC20/ERC20Burnable.sol
- @openzeppelin/contracts/access/Ownable.sol

Inheritance

SigToken contract is ERC20, Ownable, ERC20Burnable.

Usages

SigToken contract has no custom usages.

Structs

SigToken contract has following data structures:

- struct Checkpoint - stores votes checkpoints.

Enums

SigToken contract has no custom enums.

Events

SigToken contract has following custom evets:

- event DelegateChanged(address indexed delegator, address indexed fromDelegate, address indexed toDelegate)
- event DelegateVotesChanged(address indexed delegate, uint256 previousBalance, uint256 newBalance)

Modifiers

SigToken has no custom modifiers.

Fields

SigToken contract has following fields and constants:

- uint256 constant MAX_MAIN_SUPPLY = 21_000_000 * 1e18
- uint256 constant BONUS_SUPPLY = 3_024_000 * 1e18
- uint256 constant MAX_TOTAL_SUPPLY = MAX_MAIN_SUPPLY + BONUS_SUPPLY
- uint256 public startBlock
- uint256 constant DECIMALS_MUL = 1e18
- uint256 constant BLOCKS_PER_WEEK = 40_320
- uint256 constant HALVING_BLOCKS = 210_000
- mapping (address => address) internal _delegates
- mapping (address => mapping (uint32 => Checkpoint)) public checkpoints
- mapping (address => uint32) public numCheckpoints
- bytes32 public constant DOMAIN_TYPEHASH = keccak256("EIP712Domain(string name,uint256 chainId,address verifyingContract)")
- bytes32 public constant DELEGATION_TYPEHASH = keccak256("Delegation(address delegatee,uint256 nonce,uint256 expiry)")

- mapping (address => uint) public nonces

Functions

SigToken has following public functions:

- ***constructor***
  Description
  Deploys the contract. Mints a *_tinyMint* amount of token to the contract itself.
  Visibility
  public
  Input parameters
  - uint256 _tinyMint - an initial token supply.
  Constraints
  None
  Events emit
  None
  Output
  None
- ***delegate***
  Description
  Delegate votes from *msg.sender* to *delegate*.
  Visibility
  public
  Input parameters
  - address delegatee
  Constraints
  None
  Events emit
  Emits *DelegateChanged* event.
  Output
  None
- ***delegateBySig***
  Description
  Delegates votes from signatory to *delegatee*.
  Visibility
  public
  Input parameters
  - address delegate
  - uint256 nonce
  - uint256 expiry
  - uint8 v
  - bytes32 r
  - bytes32 s
  Constraints
  None

Events emit
Emits *DelegateChanged* event.
Output
None

- ***getCurrentVotes***
  Description
  Get current votes balance for *account*.
  Visibility
  external view
  Input parameters
    - address account
  Constraints
  None
  Events emit
  None
  Output
    - uint256 – number of current votes for *account*.
- ***getPriorVotes***
  Description
  Determine the prior number of votes for an *account* as of a *blockNumber*.
  Visibility
  public view
  Input parameters
    - address account
    - uint256 blockNumber
  Constraints
  None
  Events emit
  None
  Output
    - uint256 – number of votes the account had as of the given block.
- ***mint***
  Description
  Mints an *_amount* to *_to* address.
  Visibility
  public
  Input parameters
    - address _to
    - uint256 _amount
  Constraints
    - *onlyOwner* modifier.
  Events emit
  None

Output
None

SigThreePoolProxy.sol

Description

SigThreePoolProxy is a proxy for ThreePool.vy. Also contains cashback logic.

Imports

SigThreePoolProxy contract has following imports:

- @openzeppelin/contracts/token/ERC20/ERC20.sol
- @openzeppelin/contracts/token/ERC20/ERC20.sol
- @openzeppelin/contracts/math/Math.sol
- ./SigToken.sol

Inheritance

SigThreePoolProxy contract does not inherit anything.

Usages

SigThreePoolProxy contract has no custom usages.

Structs

SigThreePoolProxy contract has no data structures.

Enums

SigThreePoolProxy contract has no custom enums.

Events

SigThreePoolProxy contract has no evets.

Modifiers

SigThreePoolProxy has no custom modifiers.

Fields

SigThreePoolProxy contract has following fields and constants:

- uint256 constant N_COINS = 3;
- address[] public coins;
- uint256[] public balances;
- uint256 constant FEE_DENOMINATOR = 10* 10 ** 9;
- uint256 constant MgAX_ADMIN_FEE = 10 * 10 ** 9; // 1%

- uint256 constant MAX_FEE = 5 * 10 ** 9; // 0.5%
- uint256 public fee;
- uint256 public admin_fee;
- address public owner;
- address token;
- uint256 public initial_A;
- uint256 public future_A;
- uint256 public initial_A_time;
- uint256 public future_A_time;
- uint256 public admin_actions_deadline;
- uint256 public transfer_ownership_deadline;
- uint256 public future_fee;
- uint256 public future_admin_fee;
- address public future_owner;
- uint64 public slot_fill_0;
- uint32 public slot_fill_1;
- bool is_killed;
- uint256 kill_deadline;
- uint256 constant KILL_DEADLINE_DT = 2 * 30 * 86400;
- address delegationTarget;
- address dutchAuction;
- SigToken sigToken;
- uint256 constant CASHBACK_EXCHANGE_THRESHOLD_0 = 100e18; // at least 100 stablecoins to get reward
- uint256 constant CASHBACK_SIG_LOW_AMOUNT = 30e18;
- uint256 constant CASHBACK_SIG_HIGH_AMOUNT = 100e18;
- uint256 constant CASHBACK_MAX_GAS_PRICE = 100e9; // 100 gwei
- uint256 startBlock;
- uint256 endBlock;
- int256 a;
- int256 c;

Functions

SigThreePoolProxy has following public functions:

- ***constructor***
  Description
  Deploys the contract.
  Visibility
  public
  Input parameters
  - address _owner,

- address[N_COINS] memory _coinsIn
- address _pool_token
- uint256 _A
- uint256 _fee
- uint256 _admin_fee
- address _delegationTarget
- address _dutchAuction
- SigToken _sigToken

Constraints
None
Events emit
None
Output
None

- ***fallback***
  Description
  Delegates a call to the contract deployed at delegationTarget address.
  Visibility
  external
  Input parameters
  None
  Constraints
  None
  Events emit
  None
  Output
  None
- ***withdraw_admin_fees***
  Description
  Withdraw fees to the dutchAuction address.
  Visibility
  public
  Input parameters
  None
  Constraints
  None
  Events emit
  None
  Output
  None
- ***changeAuction***
  Description
  Change the dutchAuction value.
  Visibility
  public

Input parameters
- address newAuction

Constraints
- Can only be called by the owner

Events emit
None

Output
None

- ***setCashbackEther***

Description
Acceps ETH and sets parameters used in cashback calculations.

Visibility
public payable

Input parameters
- uint256 _startBlock
- uint256 _endBlock
- int256 initCashback
- int256 totalCashback

Constraints
- Can only be called by the owner
- The contract balance should be greater or equal to totalCashback

Events emit
None

Output
None

- ***exchange2***

Description
Calls the *exchange* function of the *ThreePool* and trying to pay a cashback.

Visibility
public

Input parameters
- int128 i
- int128 j
- uint256 dx
- uint256 min_dy

Constraints
None

Events emit
None

Output
None

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Audit overview

Critical

No critical issues were found.

High

No high severity issues were found.

Medium

1. The *add* function of the *SigMasterChef* contract is lack of validations for the *_lpToken* existence.
2. *setCashbackEther*, *availableCashbackEther*, *entitledCashbackEther*, *payCashbackEther*, *calcCashbackEther* functions of the *SigThreePoolProxy* contains unsafe math operations.

Low

No low severity issues were found.

Lowest / Code style / Best Practice

1. A lot of code style issues were found by the static code analyzers.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. For the contract, high-level description of functionality was presented in As-Is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security engineers found **2** medium, **1** informational issue during the audit.

Violations in the following categories were found and addressed to Customer:

Category	Check Item	Comments
Code review	■ Style guide violation	■ The conde contains a lot of commented out blocks and a lot of useless comments

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. Thus, the audit can't guarantee the explicit security of the audited smart contracts.