

HOLOCLEAR

ecosystem

Security Assessment & Correctness

October 29th, 2022

Audited By:

Angelos Apostolidis

angelos.apostolidis@ouovoros.io

Sheraz Arshad

sheraz.arshad@ouovoros.io

Overview

Project Summary

Project Name	HOLOCLEAR - ecosystem
Website	HOLOCLEAR
Description	An automated yielding platform
Platform	BSC; Vyper

Audit Summary

Delivery Date	October 29th, 2022
Method of Audit	Static Analysis, Manual Review

Vulnerability Summary

Total Issues	38
● Total Major	7
● Total Minor	12
● Total Informational	19

Files In Scope

Contract

contracts/MarketOracle.vy

contracts/Lottery.vy

contracts/Vault.vy

contracts/Gilts.vy

contracts/LPManager.vy

contracts/qHOLO.vy

contracts/HoloClear.vy

contracts/Swap.vy

contracts/PresaleTransfer.vy

contracts/HoloYield.vy

contracts/Escrow.vy

contracts/Bank.vy

contracts/HoloWrap.vy

interfaces/VRFCoordinatorV2.vy

interfaces/IFactory.vy

interfaces/IRouter.vy

interfaces/IPair.vy

Findings

ID	Title	Type	Severity
F-01	Unlocked Compiler Version	Language Specific	informational
F-02	Redundant Variable Initialization	Language Specific	informational
F-03	Inexistent Input Sanitization	Volatile Code	minor
F-04	Approved Transfer Extension	Gas Optimization	informational
F-05	Redundant State Look-Up	Gas Optimization	informational
F-06	Race Condition	Volatile Code	minor
F-07	Potential `event` Emission	Off-Chain Tracking	informational
F-08	Mutability Specifiers Missing	Gas Optimization	informational
F-09	`event` Optimization	Language Specific	informational
F-10	Inexistent Access Control	Logical Issue	major
F-11	Redundant `event`	Inconsistency	major
F-12	Unused Returned Value	Inconsistency, Coding Style	informational
F-13	Redundant State Variable	Gas Optimization	informational
F-14	Privileged Presale Ownership	Centralization	minor
F-15	Privileged Vault Ownership	Centralization	minor
F-16	Privileged Token Ownership	Centralization	minor
F-17	Privileged Gilts Ownership	Centralization	minor
F-18	Redundant Calculation	Gas Optimization	informational
F-19	Presale Claim Calculation	Mathematical Operations	informational
F-20	Redeemed Claim Inconsistency	Volatile Code	informational
F-21	Ambiguous `event` Emission	Volatile Code	minor
F-22	Untracked Stake	Volatile Code	major
F-23	Unused Local Variable	Gas Optimization	informational

F-24	Code Optimization	Gas Optimization	informational
F-25	Missing `event`	Off-Chain Tracking	major
F-26	Ambiguous State Variable	Off-Chain Tracking	informational
F-27	Struct Optimization	Gas Optimization	informational
F-28	State Layout Optimization	Gas Optimization	informational
F-29	Potential Re-Entrancy	Volatile Code	minor
F-30	Locked Ether	Volatile Code	major
F-31	Max Fee Feature	Volatile Code	informational
F-32	Ambiguous Math	Mathematical Operations	minor
F-33	Consecutive Winner	Logical Issue, Volatile Code	major
F-34	Requisite Value of ERC-20 `transferFrom()` / `transfer()` Call	Logical Issue	minor
F-35	Contract Initializing	Volatile Code	major
F-36	Permissionless AMM Interaction	Volatile Code	minor
F-37	Potential Sandwich Attack	Volatile Code	minor
F-38	Third Party Dependencies	Control Flow	informational

F-01: Unlocked Compiler Version

Type	Severity	Location
Language Specific	● informational	[General]

Description:

The contract has unlocked compiler version. An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers. This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation:

We advise that the compiler version is instead locked at the lowest version possible that the contract can be compiled at. For example, for version `v0.3.7` the contract should contain the following line:

```
# @version 0.3.7
```

Alleviation:

The team opted to consider our references and locked the compiler to version `0.3.7`.

F-02: Redundant Variable Initialization

Type	Severity	Location
Language Specific	● informational	[LPManager L26]; [HoloClear L103]; [MarketOracle L34]

Description:

All variable types within Solidity/Vyper are initialized to their default 'empty' value, which is usually their zeroed out representation.

Recommendation:

We advise that the linked initialization statements are removed from the codebase to increase legibility.

Alleviation:

The team opted to consider our references and removed the redundant code.

F-03: Inexistent Input Sanitization

Type	Severity	Location
Volatile Code	● minor	[Bank L28, L37]; [Escrow L22-L23, L32]; [Gilts L126, L135]; [HoloClear L161, L183, L280, L287, L305, L312, L321, L328, L408]; [HoloWrap L63, L71, L80, L118]; [HoloYield L140, L287, L298, L313, L323, L335, L343]; [LPManager L29, L36]; [qHOLON L68, L78, L92]; [Vault L181, L198]

Description:

The linked assignments arbitrarily use the input address without prior checks to its values.


Recommendation:

We advise adding an `assert` statement, checking the input values of the arguments.

Alleviation:

The team opted to consider our references and added the recommended `assert` statements to the majority of the linked functions with the exception of the `qHOLON` contract.

F-04: Approved Transfer Extension

Type	Severity	Location
Gas Optimization	 informational	[HoloClear L287, L408]; [HoloWrap L71, L118]; [HoloYield L298]; [qHOLO L78]

Description:

The `transferFrom()` function can be extended to save gas when a user has unlimited approval.


Recommendation:

We advise checking the allowance the caller has from the token owner and, unless it's the maximum number of the `uint256` type, decrement the caller's allowance.

Alleviation:

The team opted to consider our references and added the described feature to the aforementioned contracts with the exception of the `qHOLO` contract.

F-05: Redundant State Look-Up

Type	Severity	Location
Gas Optimization	 informational	[HoloClear L241]; [HoloYield L262-L263]; [qHOLo L82]

Description:

The linked assignments redundantly uses state variables.

Recommendation:

We advise utilizing local variables or function parameters instead.

Alleviation:

The team has acknowledged this exhibit but decided to use their existing implementation.

F-06: Race Condition

Type	Severity	Location
Volatile Code	● minor	[HoloWrap L80]; [qHOLo L92]

Description:

The token suffers from a known race conditional where:

- Alice allows Bob to transfer N of Alice's tokens ($N > 0$) by calling approve method on Token smart contract passing Bob's address and N as method arguments.
- After some time, Alice decides to change from N to M ($M > 0$) the number of Alice's tokens Bob is allowed to transfer, so she calls approve method again, this time passing Bob's address and M as method arguments.
- Bob notices Alice's second transaction before it was mined and quickly sends another transaction that calls transferFrom method to transfer N Alice's tokens somewhere.
- If Bob's transaction will be executed before Alice's transaction, then Bob will successfully transfer N Alice's tokens and will gain an ability to transfer another M tokens.
- Before Alice noticed that something went wrong, Bob calls transferFrom method again, this time to transfer M Alice's tokens.

So, Alice's attempt to change Bob's allowance from N to M ($N > 0$ and $M > 0$) made it possible for Bob to transfer $N+M$ of Alice's tokens, while Alice never wanted to allow so many of her tokens to be transferred by Bob.


Recommendation:

We advise to implement `increaseAllowance()` and `decreaseAllowance()` functions to update the allowance of a user and only use `approve()` when the allowance is zero.

Alleviation:

The team opted to consider our references and implemented `increaseAllowance()` and `decreaseAllowance()` functions with the exception of the `qHOLo` contract.

F-07: Potential event Emission

Type	Severity	Location
Off-Chain Tracking	 informational	[Bank L28, L37]; [LPManager L29]; [PresaleTransfer L99, L122]; [qHOLo L58]

Description:

The linked functions are part of the core functionality from a user's perspective.

Recommendation:

We advise adding events in the aforementioned functions to allow greater off-chain tracking of the system.

Alleviation:

The team opted to consider our references and added more event s with the exception of the qHOLo contract.

F-08: Mutability Specifiers Missing

Type	Severity	Location
Gas Optimization	 informational	[Bank L14]; [Escrow L14]; [Gilts L60]; [HoloClear L47-L49, L55]; [HoloWrap L30-L34]; [HoloYield L64-L69, L80]; [LPManager L20]; [Lottery L65, L68, L72, L77]; [MarketOracle L27]; [PresaleTransfer L36]; [qHOLO L31-L33, L39]; [Swap L31]; [Vault L95]

Description:

The linked state variables are assigned to only once, during the `constructor`'s execution.

Recommendation:

We advise that the `immutable` mutability specifier is set at the variable's contract-level declaration to greatly optimize the gas cost of utilizing the variables.

Alleviation:

The team opted to consider our references and changed the mutability of more state variables to `immutable` with the exception of the `qHOLO` contract.

F-09: event Optimization

Type	Severity	Location
Language Specific	● informational	[Bank L10-L12]

Description:

The linked event declaration can add their address parameters on the topics data structure, allowing easier off-chain tracking.

Recommendation:

We advise to use the indexed attribute on the linked event argument.

Alleviation:

The team opted to consider our references and added the indexed keyword to the aforementioned event declaration.

F-10: Inexistent Access Control

Type	Severity	Location
Logical Issue	● major	[HoloWrap L54-L57]

Description:

The HoloWrap implementation incorporates a `set_holoyield()` function meant to be invoked to change the yield token address this contract interacts with, while having no form of access control.

Recommendation:

We advise some form of access control to be imposed here as the linked function would enable anyone to change the yield token address, pointing to a malicious smart contract with unintended functionality.

Alleviation:

The team opted to consider our references and restricted the aforementioned functionality to be invocable only by the contract owner.

F-11: Redundant event

Type	Severity	Location
Inconsistency	● major	[qHOLLO L25-L27]

Description:

The existence of the `Mint` event can be redundant.


Recommendation:

We advise the aforementioned `event` is omitted and its invocations can be changed with `Transfer` ones.

Alleviation:

The team decided not to support the `qHOLLO` contract on mainnet.

F-12: Unused Returned Value

Type	Severity	Location
Inconsistency, Coding Style	 informational	[HoloClear L217, L282, L289, L290, L307, L314, L323, L403, L412]; [HoloYield L278]; [Vault L171]

Description:

The linked invocation does not check the return value of the `rebase()` function call.


Recommendation:

We advise the returned variables are either utilized or omitted from the function declaration.

Alleviation:

The team opted to consider our references and removed the returned values from the internal functions that were not required.

F-13: Redundant State Variable

Type	Severity	Location
Gas Optimization	 informational	[PresaleTransfer L40]

Description:

The `qho1o` state variable remains unused throughout the codebase.

Recommendation:

We advise removing the redundant state.

Alleviation:

The team opted to consider our references and removed the `qho1o` state variable.

F-14: Privileged Presale Ownership

Type	Severity	Location
Centralization	● minor	[PresaleTransfer L69, L83, L90]

Description:

The `owner` of contract has the permission to:

- set the vesting period,
- set the vesting fraction and
- set the conversion factor

while the presale is live.

Recommendation:

We advise renouncing ownership immediately after the presale is initialized.

Alleviation:

The team has acknowledged this exhibit, commenting that a centrally managed system is the intended functionality.

F-15: Privileged Vault Ownership

Type	Severity	Location
Centralization	● minor	[Vault L132, L139, L271]

Description:

The `owner` of contract has the permission to:

- set the escrow address,
- set the escrow period and
- arbitrarily withdraw collateral tokens

Such powers can block the users from withdrawing their staked assets, while the contract is drained from the staked assets.

Recommendation:

We advise migrating to a timelock governing procedure for the vault contract.

Alleviation:

The team has acknowledged this exhibit, commenting that a centrally managed system is the intended functionality.

F-16: Privileged Token Ownership

Type	Severity	Location
Centralization	● minor	[HoloClear L111, L118, L125, L132, L161, L168]

Description:

The `owner` of contract has the permission to:

- set the buy and sell fees,
- set the token swap limit and
- set the `vault`, `bank` and `gilts` addresses

without obtaining the consensus of the community. Such powers can make the owner the sole minter of the token.

Recommendation:

We advise migrating to a timelock governing procedure for the token contract.

Alleviation:

The team has acknowledged this exhibit, commenting that a centrally managed system is the intended functionality.

F-17: Privileged Gilts Ownership

Type	Severity	Location
Centralization	● minor	[Gilts L126, L135]

Description:

The `owner` of contract has the permission to set the different gilt parameters after its initialization, such as gilt bonus and gilt status (live or not). Such powers can mislead the users into participating in the gilts schema, while it can change over time.

Recommendation:

We advise migrating to a timelock governing procedure for the vault contract.

Alleviation:

The team has acknowledged this exhibit, commenting that a centrally managed system is the intended functionality.

F-18: Redundant Calculation

Type	Severity	Location
Gas Optimization	● informational	[PresaleTransfer L116]

Description:

The linked statement redundantly calculates the available yield, as the statement in [L111] already contains the value.

Recommendation:

We advise storing the result of the external call to the yield token contract in a local variable and utilizing that local variable to update the `Claim` mapping.

Alleviation:

The team opted to consider our references and introduced a local variable to reduce the storage look-ups.

F-19: Presale Claim Calculation

Type	Severity	Location
Mathematical Operations	● informational	[PresaleTransfer L152-L154]

Description:

The `_multiplier` and `_send_amount` local variables correlate with non-immutable state variables, i.e. the vesting's interval and fraction. A change in the aforementioned state variables can lead to broken arithmetics in the system.

Recommendation:

We advise renouncing ownership immediately after the presale is initialized.

Alleviation:

The team opted to consider our references and added the `renounce_ownership()` function to remove the owner.

F-20: Redeemed Claim Inconsistency

Type	Severity	Location
Volatile Code	● informational	[PresaleTransfer L171]

Description:

As explained in the previous exhibit, an edge case appears whereby the presale contract's owner can change either the vesting interval or its fraction, while users exploit the said change to claim more tokens than intended. Such claims will be processed as 'partials' and will be marked as 'unredeemed'.

Recommendation:

We advise renouncing ownership immediately after the presale is initialized.

Alleviation:

The team opted to consider our references and added the `renounce_ownership()` function to remove the owner.

F-21: Ambiguous event Emission

Type	Severity	Location
Volatile Code	● minor	[PresaleTransfer L156]

Description:

The internal `_amount_claimable()` function emits a `Redemption` event, while being invocable from the `external view` function `redeemable()`.

Recommendation:

We advise that the `Redemption` event emission is moved to the `claim()` function.

Alleviation:

The team opted to consider our references but decided to remove the `Redemption` event from the contract altogether.

F-22: Untracked Stake

Type	Severity	Location
Volatile Code	● major	[Vault L179-L195]

Description:

The vault contract does not keep track of the collateral the user provided, but is based entirely on the yields. Such an approach cannot evaluate on-chain the user's profit/loss.


Recommendation:

We advise revising the staking schema.

Alleviation:

The team opted to consider our references and added a state variable to keep track of how much H0LO is added/removed with each stake/unstake for a user.

F-23: Unused Local Variable

Type	Severity	Location
Gas Optimization	 informational	[Vault L202]

Description:

The `_balance` variable remains unused throughout the `update_lottery()` function.


Recommendation:

We advise removing redundant code.

Alleviation:

The team opted to consider our references and remove the redundant code.

F-24: Code Optimization

Type	Severity	Location
Gas Optimization	 informational	[Vault L252]

Description:

The `expiry_time` variable is only be utilized if the `escrowPeriod` state variable greater than zero.

Recommendation:

We advise moving the linked statement inside the `if` clause.

Alleviation:

The team opted to consider our references and moved the local variable inside the `if` clause.

F-25: Missing event

Type	Severity	Location
Off-Chain Tracking	● major	[HoloClear L277]; [HoloClear L410]; [HoloWrap L71, L118]

Description:

The `_transfer()` function in the `HoloClear` contract does not emit a `Transfer` event after the transfer fee is sent to the contract. Also, the `Transfer` event emitted on [L250] is misleading, as the amount the user receives will not always be the input amount (due to fee deduction). Lastly, the `transferFrom()` and `burnFrom()` functions in the `HoloWrap` contract and `burnFrom()` from `HoloClear` do not emit `Approval` events. This can lead to inconsistencies to the off-chain services that keep track of the on-chain data.

Recommendation:

We advise emitting two `Transfer` events, one for the transferred amount and one for the fee.

Alleviation:

The team opted to consider our references and added the correct `event` emissions.

F-26: Ambiguous State Variable

Type	Severity	Location
Off-Chain Tracking	● informational	[HoloClear L316]; [HoloYield L328]

Description:

The `has_interacted` state variable is ambiguously updated only via the `approve_max()` function. The state change can lead to inconsistency, as the user can change the allowance to 'infinity' via `approve()` and `increaseAllowance()` functions along with the `approve_max()` one.

Recommendation:

We advise updating the `has_interacted` map in all the relevant places.

Alleviation:

The team opted to consider our references and updated the aforementioned state variable in the proper functions.

F-27: Struct Optimization

Type	Severity	Location
Gas Optimization	● informational	[Gilts L36, L40]; [Lottery L54]

Description:

The `Claim` struct redundantly stores the `_id` member of a `Claim`, as its value is fetched from other sources in the state. Also, the `Gilt` & `WinnerInfo` structs can be optimized by tightly packing their members.

Recommendation:

We advise removing the `_id` member of the `Claim` struct. Lastly, consider grouping `market_address` & `live` member of the `Gilt` struct and `_address` & `_claimed` member of the `WinnerInfo` struct.

Alleviation:

The team opted to consider our references and moved the `struct` members to strive for a tight packing. Also, they commented that the aforementioned `_id` struct member is needed for the UI.

F-28: State Layout Optimization

Type	Severity	Location
Gas Optimization	● informational	[HoloClear L55, L81-L83]

Description:

The state of the `HoloClear` contract is not tightly packed in 256-bit slots.

Recommendation:

We advise to move the `owner` state variable adjacent to the `liquify_enabled` & `swap_locked` ones, striving for 256-bit packing.

Alleviation:

The team opted to consider our references and moved the state variable layout to strive for a tight packing.

F-29: Potential Re-Entrancy

Type	Severity	Location
Volatile Code	● minor	[HoloWrap L126];

Description:

The linked code segments update the state of their respective contract after an external call.

Recommendation:

We advise to execute the external call at the end of the function, hence following the [Checks-Effects-Interactions pattern](#).

Alleviation:

The team opted to consider our references and added the `@nonreentrant` decorator to lock the `wrap` and `unwrap` functions from re-entrancy.

F-30: Locked Ether

Type	Severity	Location
Volatile Code	● major	[Lottery L94-L97]; [Swap L39-L42]

Description:

The contract does not have withdrawal capacity, while it is possible to hold Ether via its `__default__()` function.

Recommendation:

We advise to either add a withdraw function to avoid having Ether permanently locked in the contract or remove the `__default__()` function.

Alleviation:

The team opted to consider our references and added the `withdraw()` and `withdraw_quote()` functions to both contracts.

F-31: Max Fee Feature

Type	Severity	Location
Volatile Code	● informational	[Swap L63]

Description:

The linked setter function allows for arbitrary fee value.

Recommendation:

We advise adding an upper bound to the fee value.

Alleviation:

The team opted to consider our references and added a max fee of ten percent (10%).

F-32: Ambiguous Math

Type	Severity	Location
Mathematical Operations	● minor	[MarketOracle L95-L96]

Description:

There is a small deviation from the arithmetics the interacting AMM uses when the reserves are higher than the max value of the `uint144` type.

Recommendation:

We advise following the arithmetics of the AMM.

Alleviation:

The team has acknowledged this exhibit but decided to use their existing implementation.

F-33: Consecutive Winner

Type	Severity	Location
Logical Issue, Volatile Code	● major	[Lottery L235]

Description:

If the same address is chosen as the winner of a second, consecutive, draw and the winner had not claimed the rewards from the previous one, then the winning amount is reset to current winning, thus rendering the previous unclaimable.

Recommendation:

We advise revising the lottery draw winner and claim functionality from contract.

Alleviation:

The team opted to consider our references and changed the lottery winner drawing logic to disallow for an address to win two consecutive times.

F-34: Requisite Value of ERC-20 `transferFrom()` / `transfer()` Call

Type	Severity	Location
Logical Issue	● minor	[Bank L43]; [Gilts L199-L200]; [Gilts L117, L121, L153-L155]

Description:

While the ERC-20 implementation does necessitate that the `transferFrom()` / `transfer()` function returns a `bool` variable yielding `true`, many token implementations do not return anything i.e. Tether (USDT) leading to unexpected halts in code execution.

Recommendation:

We advise utilizing the `raw_call` built-in function to ensure that the `transferFrom()` / `transfer()` function is safely invoked in all circumstances.

Alleviation:

The team has acknowledged this exhibit but decided to use their existing implementation.

F-35: Contract Initializing

Type	Severity	Location
Volatile Code	● major	[Escrow L22]; [Gilts L94]; [HoloYield L119]; [Lottery L111]; [PresaleTransfer L59]; [Swap L53]; [Vault L113]

Description:

The linked initializing function can be invoked more than once.

Recommendation:

We advise to add a 'use only once' guard to each linked function.

Alleviation:

The team opted to consider our references and refactored all the contract initializing functions to be invocable only once.

F-36: Permissionless AMM Interaction

Type	Severity	Location
Volatile Code	● minor	[Swap L115, L135]

Description:

Interacting with a permissionless AMM like PancakeSwap has some hidden dangers, as malicious tokens can be blended in liquidity pools (LPs) along with normal ones.

Recommendation:

We advise making a due diligence on the supported liquidity pools (LPs).

Alleviation:

The team has acknowledged this exhibit.

F-37: Potential Sandwich Attack

Type	Severity	Location
Volatile Code	● minor	[Gilts L183; [HoloClear L226]; [Swap L84, L98, L129, L145]

Description:

Potential sandwich attacks could happen when calling liquidity pool based swapping functionality without slippage control.

Recommendation:

We recommend using an oracle to get an estimation of prices and setting minimum amounts based on the prices when calling the such functions.

Alleviation:

The team has acknowledged this exhibit.

F-38: Third Party Dependencies

Type	Severity	Location
Control Flow	● informational	[Gilts L183]; [HoloClear L226]; [Lottery L193]; [Swap L84, L98, L129, L145]

Description:

The contract is serving as the underlying entity to interact with third party protocol(s). We treat those third party entities as black boxes and assume its functional correctness. However in the real world, third parties may be compromised that led to assets lost or stolen.

Recommendation:

We encourage the team to constantly monitor the statuses of those third parties, to mitigate the side effects when unexpected activities are observed.

Alleviation:

The team has acknowledged this exhibit.

Disclaimer

Reports made by Ourovoros are not to be considered as a recommendation or approval of any particular project or team. Security reviews made by Ourovoros for any project or team are not to be taken as a depiction of the value of the “product” or “asset” that is being reviewed.

Ourovoros reports are not to be considered as a guarantee of the bug-free nature of the technology analyzed and should not be used as an investment decision with any particular project. They represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Each company and individual is responsible for their own due diligence and continuous security. Our goal is to help reduce the attack parameters and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claim any guarantee of security or functionality of the technology we agree to analyze.