CERTIK

Security Assessment

Golff Vault

Apr 25th, 2021

Summary

This report has been prepared for Golff Vault smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Dynamic Analysis, Static Analysis, and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Golff Vault
Description	One-Stop Services Encrypted Bank
Platform	BSC
Language	Solidity
Codebase	https://github.com/golfffinance/golff-bsc-vault/tree/master/contracts
Commits	1be8385cc542eda56527946407bba539f22f31f1

Audit Summary

Delivery Date	Apr 25, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Total Issues	25
Critical	0
Major	4
Minor	3
Informational	18
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
GVE	GofVault.sol	3717751f0387e4e3adeac6a38d94d60598abdbc4f1d2fcf959c18043ffc467f1
GVB	GofVaultBNB.sol	df5349ff29008464706bb803bc0085efdc4a773b53e48110b526358d7ff27ba3
MIG	Migrations.sol	9caeb6c8cd59529581eddc108dd7b67766720f88ae97255b8297e5127c713503
GOF	controller/GOFControllerV1.sol	a6b32acc532b2162c9b488758d237f4a2f165f43b4d89ae5876990f12d9d936e
IER	interfaces/IERC20Detailed.sol	b6a54a4c16be5b9cf13983585f341339f00fbf78bb75a80f92a13f864b6fdc44
IGO	interfaces/IGOFController.sol	9730ddb445d0c242317ca2456cd82655fdb69cfb2fde4f19b097f7ef9ceb8a06
IGF	interfaces/IGOFPool.sol	24700b7d45edf75671b3fb9c52def78d1e245032c31eb400318066151e3a2287
IGS	interfaces/IGOFStrategy.sol	369ccd8c9624e4f00a10c25abc1f6262cd670df4f3dad8c79ebb2183a4a144f5
IGV	interfaces/IGOFVaultMigrateable.sol	8074999115179cf289a717632f5910d47e9571bcb8a46c88c450016747b14689
ISR	interfaces/ISwapRouter.sol	08acf0f6e079832ef0833eaec85f0c80e91ad4d180e723aadabfd72994906222
IWB	interfaces/IWBNB.sol	9e3b1a2b9195ec3e8131ca014857701f76b5783c5cfa1e07a2442bbfa7d5f105
IAP	interfaces/auto/IAutoPool.sol	8ac77cd1a7b8be187aa61a7ab3ff935bb93585d9a1b6c5b184c1298ebb2e1433
ICP	interfaces/cake/ICakePool.sol	88a9bf3fb82537ddc7bc33d4fb75dc7660e00ac3975672d540a029f14db35609
IMP	interfaces/mdex/IMdexPool.sol	194007de8e32bb48756f36d9ddab1a6218cc4858a33c99921c1bd1c520487cd6
IUE	interfaces/venus/IUnitroller.sol	d40ef6f70eb9dd4cd0225b5c7fd3356f3e195fe8eb5696dea92d5d0cb0a3baad

ID	file	SHA256 Checksum
IVB	interfaces/venus/IVenusBNBPool.sol	41ba715a2683ff81aee1be872275dcdaf96a69f9678697a66eb83016d0a8179c
IVP	interfaces/venus/IVenusPool.sol	60429bb3dbfc62ec88e334546e5635a0cefab80de9808f5640b49e5e12453ad7
SFA	strategy/StrategyForAuto.sol	4c78522dd1a6fd1c68a4a00c075ba0c4bcc224b69942390b2682001e57e9e595
SFC	strategy/StrategyForCakeLp.sol	9d993a9e43c7faa8e5bcc8000dde50681a29c07ec08c43d8badd800d8f7d78fd
SFM	strategy/StrategyForMdex.sol	3ab9bac0c7b7bd6adeea2eb1eeb261122654c433f7ff1b6cb27f47760e24a2da
SFL	strategy/StrategyForMdexLp.sol	fa2bb60b7e7e3975b7452c4cacc4430d06bcdb664f84372f1f2583213746ad61
SFV	strategy/StrategyForVenus.sol	7618cd643111789f5c2a2a3b852247ebd10210768500c13955fb1a3b9cf2c961
MTE	test/MockToken.sol	9593a882cd4ff507f6df9fc5e695203958eff43374edd05c60d625f9e0cc9897
WBN	test/WBNB.sol	df83ee2286ad908618ebe918b91fae7f9f10d0065260f0ce30ce8c994827ed31

System Overview

Golff Vault is an One-Stop Encrypted Bank. The core components are the Controller,Strategy and Vault, which allow users to deposit their digital assets. These assets are transferred to a third party service Pancake. In exchange, users are issued LP tokens that represent their claim on their deposits. Users will be incentivized for earning staking rewards in pool, and saving gas by aggregation staking. This protocol has external dependencies. All user deposits are immediately transferred to a third-party service (like pancakeswap or Gof Pool). The system should only be used if the service is appropriately trusted. And these external protocols are not in the scope of this audit.

Centralized Risks

Additionally, to bridge the trust gap between the administrator and users, the administrator needs to express a sincere attitude with the consideration of the administrator team’s anonymousness. The administrator has the responsibility to notify users with the following capability of the administrator:

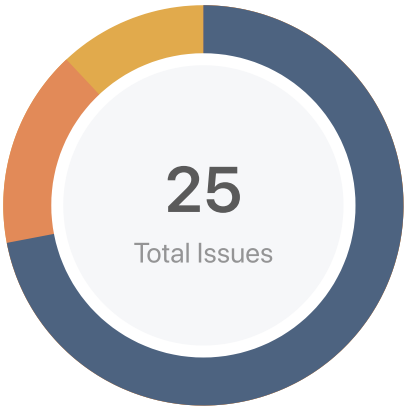
- Administrators can send assets to any address via "GOFControllerV1.inCaseTokensGetStuck()" function.

To improve the trustworthiness of the project, any dynamic runtime updates in the project should be notified to the community. Any plan to invoke the above-mentioned functions should be also considered to move to the execution queue of the Timelock contract.

Financial Models

Financial models of blockchain protocols need to be resilient to attacks. It needs to pass simulations and verifications to guarantee the security of the overall protocol.

Findings



Critical	0 (0.00%)
Major	4 (16.00%)
Minor	3 (12.00%)
Informational	18 (72.00%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
GOF-01	Proper Usage of “public” and “external” type	Coding Style	Informational	Partially Resolved
GOF-02	Boolean Equality	Gas Optimization	Informational	Resolved
GOF-03	Incorrect address	Logical Issue	Informational	Partially Resolved
GOF-04	Dangerous usage of tx.origin	Logical Issue	Major	Resolved
GOF-05	Missing Emit Events	Gas Optimization	Informational	Acknowledged
GOF-06	Redundant Codes	Coding Style	Informational	Resolved
GOF-07	Centralized Risks	**Centralization / Privilege**	**Major**	**Resolved**
GOF-08	Missing Some Important Checks	Logical Issue	Informational	Acknowledged
GOF-09	Logical Issue of `getExpectedReturn()`	Logical Issue	Major	Resolved
GVB-01	Proper Usage of “public” and “external” type	Coding Style	Informational	Partially Resolved
GVB-02	Missing Emit Events	Gas Optimization	Informational	Acknowledged
GVB-03	Issue in Receiving BNB Function	Logical Issue	Minor	Acknowledged
GVE-01	Unlocked Compiler Version Declaration	Language Specific	Informational	Acknowledged

ID	Title	Category	Severity	Status
GVE-02	Missing Some Important Checks	Logical Issue	Minor	Acknowledged
GVE-03	External Dependency	Data Flow	Minor	Acknowledged
GVE-04	Events Should Add Indexed Keyword	Language Specific	Informational	Partially Resolved
GVE-05	Missing Some Important Checks	Logical Issue	Informational	Partially Resolved
SFC-01	Simplifying Existing Code	Gas Optimization	Informational	Acknowledged
SFV-01	Incorrect Naming Convention Utilization	Coding Style	Informational	Acknowledged
SFV-02	Proper Usage of “public” and “external” type	Coding Style	Informational	Partially Resolved
SFV-03	Divide before multiply	Language Specific	Informational	Resolved
SFV-04	Incorrect address	Logical Issue	Informational	Partially Resolved
SFV-05	Missing Some Important Checks	Logical Issue	Informational	Acknowledged
SFV-06	Dangerous usage of tx.origin	Logical Issue	Major	Resolved
SFV-07	Missing Emit Events	Gas Optimization	Informational	Acknowledged

GOF-01 | Proper Usage of “public” and “external” type

Category	Severity	Location	Status
Coding Style	Informational	controller/GOFControllerV1.sol: 51, 55, 59, 63, 67, 71, 75, 79, 84, 88, 122, 126, 130, 134, 146, 176	Partially Resolved

Description

“public” functions that are never called by the contract should be declared “external” . When the inputs are arrays, “external” functions are more efficient than “public” functions.

Examples:

Functions like : `setController()`, `setEarnLowerlimit()`, `depositBehalf()`, `accrueReward()`, `getPricePerFullShare()`, `depositBNBAndFarm()`, `setStrategist()`, `setFactory()`, `setSplit()`, `setOneSplit()`, `setRewards()`, `approveStrategy()`, `revokeStrategy()`, `setVault()`, `setConverter()`, `setStrategy()`, `withdrawAll()`, `inCaseTokensGetStuck()`, `inCaseStrategyTokenGetStuck()`, `getExpectedReturn()`, `yearn()`, `withdraw()`, `deposit()`, `claimReservesAll()`, `setFees()`, `setReservesRate()`, `setFoundationAddress()`, `setWithdrawalFee()`, `setBurnAddress()`, `setStrategyDev()`, `setRouter()`, `setSwap2Token()`, `setSwap2GOF()`, `setSplitGof()`

Recommendation

Consider using the “external” attribute for functions never called from the contract.

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GOF-02 | Boolean Equality

Category	Severity	Location	Status
Gas Optimization	Informational	controller/GOFControllerV1.sol: 89	Resolved

Description

Boolean constants can be used directly and do not need to be compared to true or false.

Example:

```
require(approvedStrategies[_token][_strategy] == true, "Golff:!approved");
```

Recommendation

Consider changing it as following example:

```
require(approvedStrategies[_token][_strategy], "Golff:!approved");
```

Alleviation

The team heeded our advice and resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GOF-03 | Incorrect address

Category	Severity	Location	Status
Logical Issue	Informational	controller/GOFControllerV1.sol: 38, 42	Partially Resolved

Description

There are many incorrect addresses in the protocol.

Examples:

The address of WBNB is `0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c` instead of `0x5545153CCFcA01fbd7Dd11C0b23ba694D9509A6F`.

```
address constant public wbnb = address(0x5545153CCFcA01fbd7Dd11C0b23ba694D9509A6F);
```

The address `0x2170Ed0880ac9A755fd29B2688956BD959F933F8` is the address of ETH instead of gof.

```
address constant public gof = address(0x2170Ed0880ac9A755fd29B2688956BD959F933F8);
```

The address `0xED7d5F38C79115ca12fe6C0041abb22F0A06C300` is not a contract on the BSC

```
onesplit = address(0xED7d5F38C79115ca12fe6C0041abb22F0A06C300);
```

Recommendation

Consider to confirm the correctness of addresses used here.

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515, with the GOF team stating "The GOF token has not yet been deployed on the BSC, so the address of GOF is empty temporarily and will be added later when it is released".

GOF-04 | Dangerous usage of tx.origin

Category	Severity	Location	Status
Logical Issue	Major	controller/GOFControllerV1.sol: 41	Resolved

Description

`tx.origin` based protection can be abused by a malicious contract if a legitimate user interacts with the malicious contract.

Examples:

```
strategist = tx.origin;
```

```
strategyDev = tx.origin;
```

Recommendation

Consider to use `msg.sender`.

Alleviation

The team heeded our advice and resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GOF-05 | Missing Emit Events

Category	Severity	Location	Status
Gas Optimization	Informational	controller/GOFControllerV1.sol: 126	Acknowledged

Description

Several sensitive actions are defined without event declarations.

1. Fucntions `setController()`, `setBurnAddress()`, `setRouter()`, `setSwap2GOF()`, `setSwap2Token()` can change the governance of the contracts.
2. Fucntions `setMin()` , `available()` will decide the proportion of assest to be borrowed.
3. Function `inCaseTokensGetStuck()` can transfer any amount of assets to governance in case the controller has.
4. Fucntions `setFees()`, `setReservesRate()`, `setFoundationAddress()`, `setWithdrawalFee()`, `setStrategyDev()`, `setSplitGof()` will decide the important metrics.

Recommendation

Consider adding events for sensitive actions, and emit it in the function.

Alleviation

The recommendation was not taken into account, with the GOF team stating "They don't record log for these actions".

GOF-06 | Redundant Codes

Category	Severity	Location	Status
Coding Style	Informational	controller/GOFControllerV1.sol: 29, 55	Resolved

Description

Variable `factory` and function `setFactory()` are defined but never used.

Recommendation

We recommend removing the redundant codes.

Alleviation

The team heeded our advice and resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GOF-07 | Centralized Risks

Category	Severity	Location	Status
Centralization / Privilege	Major	controller/GOFControllerV1.sol: 126	Resolved

Description

To bridge the trust gap between the administrator and users, the administrator needs to express a sincere attitude with the consideration of the administrator team's anonymousness. The administrator has the responsibility to notify users with the following capability of the administrator:

- Administrators can transfer amount to owner address via `GOFControllerV1.inCaseTokensGetStuck()` function.

The advantage of `inCaseTokensGetStuck()` method in the protocol is that the administrator reserves the ability to rescue the assets in this contract under unexpected cases. It is also worthy of note the downside of 'inCaseTokensGetStuck' method, where the treasury in this contract can be migrated to owner address.

Recommendation

To improve the trustworthiness of the project, any dynamic runtime updates in the project should be notified to the community. Any plan to invoke the above-mentioned functions should be also considered to move to the execution queue of the Timelock contract.

Alleviation

The team heeded our advice and removed this function in commit 68d53e648202515fbed232172638519afd5396c2.

GOF-08 | Missing Some Important Checks

Category	Severity	Location	Status
Logical Issue	Informational	controller/GOFControllerV1.sol: 80	Acknowledged

Description

The value may be entered incorrectly and cannot be changed subsequently.

```
function setVault(address _token, address _vault) public checkStrategist{
        require(vaults[_token] == address(0), "Golff:vault exist");
        vaults[_token] = _vault;
    }
```

Recommendation

Consider changing it as following example:

```
function setVault(address _token, address _vault) public checkStrategist{
        require(_vault  != address(0), "_vault is zero address");
        require(vaults[_token] == address(0), "Golff:vault exist");
        vaults[_token] = _vault;
    }
```

Alleviation

The recommendation was not taken into account, with the GOF team stating "They will control by themselves".

GOF-09 | Logical Issue of `getExpectedReturn()`

Category	Severity	Location	Status
Logical Issue	Major	controller/GOFControllerV1.sol: 138	Resolved

Description

The value of `swap2TokenRouting` is must be incorrect, because `swap2TokenRouting[0]` is not assigned and its actual value is `address(0)`.

```
address[] memory swap2TokenRouting;
swap2TokenRouting[1] = wbnb;
swap2TokenRouting[2] = _want;
uint256[] memory amountsOut = ISwapRouter(onesplit).getAmountsOut(_balance,
swap2TokenRouting);
```

Recommendation

Consider changing it as following example:

```
address[] memory swap2TokenRouting;
swap2TokenRouting[0] = _token;
swap2TokenRouting[1] = wbnb;
swap2TokenRouting[2] = _want;
ISwapRouter(onesplit).swapExactTokensForTokens(_amount, 0, swap2TokenRouting,
address(this), now.add(1800));
```

Alleviation

The team heeded our advice and resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GVB-01 | Proper Usage of “public” and “external” type

Category	Severity	Location	Status
Coding Style	Informational	GofVaultBNB.sol: 71, 74, 123, 222, 263	Partially Resolved

Description

“public” functions that are never called by the contract should be declared “external” . When the inputs are arrays, “external” functions are more efficient than “public” functions.

Examples:

Functions like : `setController()`, `setEarnLowerlimit()`, `depositBehalf()`, `accrueReward()`, `getPricePerFullShare()`, `depositBNBAndFarm()`, `setStrategist()`, `setFactory()`, `setSplit()`, `setOneSplit()`, `setRewards()`, `approveStrategy()`, `revokeStrategy()`, `setVault()`, `setConverter()`, `setStrategy()`, `withdrawAll()`, `inCaseTokensGetStuck()`, `inCaseStrategyTokenGetStuck()`, `getExpectedReturn()`, `yearn()`, `withdraw()`, `deposit()`, `claimReservesAll()`, `setFees()`, `setReservesRate()`, `setFoundationAddress()`, `setWithdrawalFee()`, `setBurnAddress()`, `setStrategyDev()`, `setRouter()`, `setSwap2Token()`, `setSwap2GOF()`, `setSplitGof()`

Recommendation

Consider using the “external” attribute for functions never called from the contract.

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GVB-02 | Missing Emit Events

Category	Severity	Location	Status
Gas Optimization	Informational	GofVaultBNB.sol: 67, 78	Acknowledged

Description

Several sensitive actions are defined without event declarations.

1. Fucntions `setController()`, `setBurnAddress()`, `setRouter()`, `setSwap2GOF()`, `setSwap2Token()` can change the governance of the contracts.
2. Fucntions `setMin()` , `available()` will decide the proportion of assest to be borrowed.
3. Function `inCaseTokensGetStuck()` can transfer any amount of assets to governance in case the controller has.
4. Fucntions `setFees()`, `setReservesRate()`, `setFoundationAddress()`, `setWithdrawalFee()`, `setStrategyDev()`, `setSplitGof()` will decide the important metrics.

Recommendation

Consider adding events for sensitive actions, and emit it in the function.

Alleviation

The recommendation was not taken into account, with the GOF team stating "They don't record log for these actions".

GVB-03 | Issue in Receiving BNB Function

Category	Severity	Location	Status
Logical Issue	Minor	GofVaultBNB.sol: 269	Acknowledged

Description

In the Ethereum, `send/transfer/call` can be used for ETH transferrings. In the worst case, the fallback function can only rely on 2300 gas being available (for example when `send` or `transfer` is used), leaving little room to perform other operations except basic logging. Therefore, the callback function of the current contract is not suitable for doing much.

```
269 receive() external payable {
270     if (msg.sender != address(token)) {
271         depositBNB();
272     }
273 }
```

The same parameter is used for Binance Smart Chain. Refer to: https://github.com/binance-chain/bsc/blob/46d185b4cfed54436f526b24c47b15ed58a5e1bb/params/protocol_params.go#L38

Recommendation

Consider to test the gas consumption of below codes:

```
269 receive() external payable {
270     if (msg.sender != address(token)) {
271         depositBNB();
272     }
273 }
```

Each opcode supported by the EVM has an associated gas cost. Pay attention the gas costs aren't arbitrary. Gas costs can and will change.

Alleviation

The Golff Team replied : We have tested the `depositBNB()` function , it consumed 452,125 gas. We used `transfer` function to send BNB to `GofVaultBNB` contract, and it successfully passed the test. Hence we believe it is not necessary to remove the `depositBNB()` function in the `receive()` function. CertiK response: `depositBNB()` function consumed 452,125 gas, which exceeds the gas limit of 2300.

But the below codes will only be triggered by user transferring. If user use `call` function, it will not have gas limit.

```
    if (msg.sender != address(token)) {
        depositBNB();
    }
```

GVE-01 | Unlocked Compiler Version Declaration

Category	Severity	Location	Status
Language Specific	Informational	GofVault.sol: 2	Acknowledged

Description

The compiler version utilized throughout the project uses the "^" prefix specifier, denoting that a compiler version which is greater than the version will be used to compile the contracts.

Recommendation

It is a general practice to instead lock the compiler at a specific version rather than allow a range of compiler versions to be utilized to avoid compiler-specific bugs and be able to identify ones more easily. We recommend locking the compiler at the lowest possible version that supports all the capabilities wished by the codebase. This will ensure that the project utilizes a compiler version that has been in use for the longest time and as such is less likely to contain yet-undiscovered bugs.

GVE-02 | Missing Some Important Checks

Category	Severity	Location	Status
Logical Issue	Minor	GofVault.sol: 216, 221	Acknowledged

Description

Functions `setMigrateDist()`, `setGofPool()` on the afore-mentioned lines are missing parameter validations.

Recommendation

Consider adding below checks:

```
require(_newVault != address(0), "_newVault is zero address");
```

Alleviation

The recommendation was not taken into account, with the GOF team stating "They will control by themselves".

GVE-03 | External Dependency

Category	Severity	Location	Status
Data Flow	Minor	GofVault.sol: 116	Acknowledged

Description

This function `depositInternal()` is calling external protocols.

```
IGOFPool(gofPool).stakeBehalf(_account, _shares);
```

The function `stakeBehalf()` is not in the scope of this audit.

GVE-04 | Events Should Add Indexed Keyword

Category	Severity	Location	Status
Language Specific	Informational	GofVault.sol: 42	Partially Resolved

Description

Event definitions in contract GOFVault do not have indexed keywords.

The indexed parameters for logged events will allow you to search for these events using the indexed parameters as filters.

```
event Deposit(address payer, address account, uint256 amount);
event Withdraw(address account, uint256 amount);
event Migrate(address account, address newVault, uint256 amount);
```

Recommendation

We recommend to add the indexed keywords.

```
event Deposit(address indexed payer, address indexed account, uint256 amount);
event Withdraw(address indexed account, uint256 amount);
event Migrate(address indexed account, address indexed newVault, uint256 amount);
```

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

GVE-05 | Missing Some Important Checks

Category	Severity	Location	Status
Logical Issue	Informational	GofVault.sol: 60	Partially Resolved

Description

It is necessary to check the magnitude of the value of `_min`.

Recommendation

Consider adding below checks:

```
require(_min <= max, "_min is over max");
```

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

SFC-01 | Simplifying Existing Code

Category	Severity	Location	Status
Gas Optimization	Informational	strategy/StrategyForCakeLp.sol: 129, 137, 157	Acknowledged

Description

Consider using a modifier to replace the below same codes existing in many functions:

```
require(msg.sender == controller, "Golff:!controller");
```

Example:

Functions `withdrawAll()`, `withdraw()` in StrategyForCakeLp.sol

Recommendation

Consider changing it as following example:

```
modifier onlyController() {
       require(msg.sender == controller, "Golff:!controller");
       _;
}
```

SFV-01 | Incorrect Naming Convention Utilization

Category	Severity	Location	Status
Coding Style	Informational	strategy/StrategyForVenus.sol: 43, 44, 45, 57, 61	Acknowledged

Description

Solidity defines a naming convention that should be followed. In general, the following naming conventions should be utilized in a Solidity file:

Constants should be named with all capital letters with underscores separating words UPPER_CASE_WITH_UNDERSCORES

Refer to https://solidity.readthedocs.io/en/v0.5.17/style-guide.html#naming-conventions

Examples:

Constants like : `wbnb`, `gof`, `unitroller`, `cashMax`, `withdrawalMax`

Recommendation

The recommendations outlined here are intended to improve the readability, and thus they are not rules, but rather guidelines to try and help convey the most information through the names of things.

SFV-02 | Proper Usage of “public” and “external” type

Category	Severity	Location	Status
Coding Style	Informational	strategy/StrategyForVenus.sol: 110, 142, 329, 333, 342, 347, 351, 356, 360, 364, 368, 372, 376	Partially Resolved

Description

“public” functions that are never called by the contract should be declared “external” . When the inputs are arrays, “external” functions are more efficient than “public” functions.

Examples:

Functions like : `setController()`, `setEarnLowerlimit()`, `depositBehalf()`, `accrueReward()`, `getPricePerFullShare()`, `depositBNBAndFarm()`, `setStrategist()`, `setFactory()`, `setSplit()`, `setOneSplit()`, `setRewards()`, `approveStrategy()`, `revokeStrategy()`, `setVault()`, `setConverter()`, `setStrategy()`, `withdrawAll()`, `inCaseTokensGetStuck()`, `inCaseStrategyTokenGetStuck()`, `getExpectedReturn()`, `yearn()`, `withdraw()`, `deposit()`, `claimReservesAll()`, `setFees()`, `setReservesRate()`, `setFoundationAddress()`, `setWithdrawalFee()`, `setBurnAddress()`, `setStrategyDev()`, `setRouter()`, `setSwap2Token()`, `setSwap2GOF()`, `setSplitGof()`

Recommendation

Consider using the “external” attribute for functions never called from the contract.

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

SFV-03 | Divide before multiply

Category	Severity	Location	Status
Language Specific	Informational	strategy/StrategyForVenus.sol: 306	Resolved

Description

Performs a multiplication on the result of a division. Solidity integer division might truncate. As a result, performing multiplication before division can sometimes avoid loss of precision.

Refer to https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

Example:

```
return
cbalancePrior.mul(exchangeRate.sub(exchangeRatePrior)).div(1e18).mul(reservesRate).div(cashMax);
```

Recommendation

Consider ordering multiplication before division:

```
return
cbalancePrior.mul(exchangeRate.sub(exchangeRatePrior)).mul(reservesRate).div(1e18).div(cashMax);
```

Alleviation

The team heeded our advice and resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

SFV-04 | Incorrect address

Category	Severity	Location	Status
Logical Issue	Informational	strategy/StrategyForVenus.sol: 43	Partially Resolved

Description

There are many incorrect addresses in the protocol.

Examples:

The address of WBNB is `0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c` instead of `0x5545153CCFcA01fbd7Dd11C0b23ba694D9509A6F`.

```
address constant public wbnb = address(0x5545153CCFcA01fbd7Dd11C0b23ba694D9509A6F);
```

The address `0x2170Ed0880ac9A755fd29B2688956BD959F933F8` is the address of ETH instead of gof.

```
address constant public gof = address(0x2170Ed0880ac9A755fd29B2688956BD959F933F8);
```

The address `0xED7d5F38C79115ca12fe6C0041abb22F0A06C300` is not a contract on the BSC

```
onesplit = address(0xED7d5F38C79115ca12fe6C0041abb22F0A06C300);
```

Recommendation

Consider to confirm the correctness of addresses used here.

Alleviation

The team heeded our advice and partially resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515, with the GOF team stating "The GOF token has not yet been deployed on the BSC, so the address of GOF is empty temporarily and will be added later when it is released".

SFV-05 | Missing Some Important Checks

Category	Severity	Location	Status
Logical Issue	Informational	strategy/StrategyForVenus.sol: 84, 85, 86, 88, 89, 90, 330, 365, 357, 361, 369, 373	Acknowledged

Description

The assigned value to `_controller`, `_want`, `_output`, `_poolAddress`, `_routerAddress`, `_burnAddress`, `_strategyDev`, `_path` should be verified as non zero value to prevent being mistakenly assigned as `address(0)` in contract `StrategyForVenus.sol`.

Recommendation

Check the address is not zero by adding following checks in the constructor of contract.

Example:

```
require(_want != address(0), "_want is zero address");
```

Alleviation

The recommendation was not taken into account, with the GOF team stating "They will control by themselves".

SFV-06 | Dangerous usage of tx.origin

Category	Severity	Location	Status
Logical Issue	Major	strategy/StrategyForVenus.sol: 83	Resolved

Description

`tx.origin` based protection can be abused by a malicious contract if a legitimate user interacts with the malicious contract.

Examples:

```
strategist = tx.origin;
```

```
strategyDev = tx.origin;
```

Recommendation

Consider to use `msg.sender`.

Alleviation

The team heeded our advice and resolved this issue in commit 065ad7d0d303501ebbbd1a0742524a9a47f1d515.

SFV-07 | Missing Emit Events

Category	Severity	Location	Status
Gas Optimization	Informational	strategy/StrategyForVenus.sol: 329, 333, 333, 342, 347, 351, 356, 360, 364, 368, 372, 376	Acknowledged

Description

Several sensitive actions are defined without event declarations.

1. Fucntions `setController()`, `setBurnAddress()`, `setRouter()`, `setSwap2GOF()`, `setSwap2Token()` can change the governance of the contracts.
2. Fucntions `setMin()` , `available()` will decide the proportion of assest to be borrowed.
3. Function `inCaseTokensGetStuck()` can transfer any amount of assets to governance in case the controller has.
4. Fucntions `setFees()`, `setReservesRate()`, `setFoundationAddress()`, `setWithdrawalFee()`, `setStrategyDev()`, `setSplitGof()` will decide the important metrics.

Recommendation

Consider adding events for sensitive actions, and emit it in the function.

Alleviation

The recommendation was not taken into account, with the GOF team stating "They don't record log for these actions".

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings refer to exhibits that do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation exhibits entail findings that relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings are exhibits that detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Data Flow

Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a struct assignment operation affecting an in-memory struct rather than an in storage one.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete .

Coding Style

Coding Style findings usually do not affect the generated byte-code and comment on how to make the codebase more legible and as a result easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers

Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error

Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK's prior written consent.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.