CERTIK

Security Assessment

CatCoin V2

May 13th, 2022

Table of Contents

Summary

This report has been prepared for CatCoin V2 to discover issues and vulnerabilities in the source code of the CatCoin V2 project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing 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;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	CatCoin V2
Platform	Ethereum
Language	Solidity
Codebase	https://bscscan.com/address/0x728829B74577831F47953BCDBC8dfca27141c56e#code
Commit	

Audit Summary

Delivery Date	May 13, 2022 UTC
Audit Methodology	Static Analysis, Manual Review

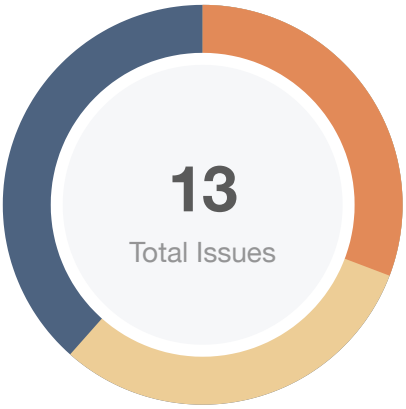
Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Mitigated	Partially Resolved	Resolved
Critical	0	0	0	0	0	0	0
Major	4	0	0	1	2	0	1
Medium	0	0	0	0	0	0	0
Minor	4	0	0	4	0	0	0
Informational	5	0	0	5	0	0	0
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	Repo	File	SHA256 Checksum
CAT	mainnet	Catcoin.sol	0ea70fb6ccaca9dcbbb4646ee124256bf04225ff62f281dadef10dc9077a539d

Findings



Critical	0 (0.00%)
Major	4 (30.77%)
Medium	0 (0.00%)
Minor	4 (30.77%)
Informational	5 (38.46%)
Discussion	0 (0.00%)

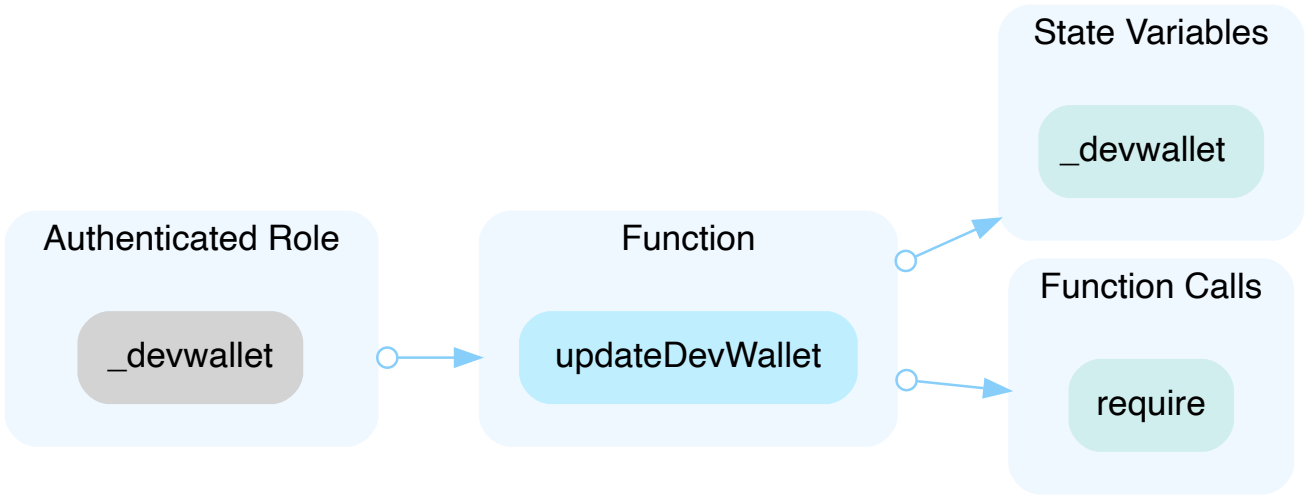
ID	Title	Category	Severity	Status
CAT-01	Centralization Risks In Catcoin.sol	**Centralization / Privilege**	**Major**	Resolved
CAT-02	Centralized Risk In `addLiquidity`	**Centralization / Privilege**	**Major**	Mitigated
CAT-03	Centralization Related Risks	**Centralization / Privilege**	**Major**	Mitigated
CAT-04	Initial Token Distribution	**Centralization / Privilege**	**Major**	Acknowledged
CAT-05	Missing Zero Address Validation	Volatile Code	Minor	Acknowledged
CAT-06	Third Party Dependencies	Volatile Code	Minor	Acknowledged
CAT-07	Potential Gas Waste In `_transfer`	Gas Optimization	Minor	Acknowledged
CAT-08	Useless Statement	Logical Issue	Minor	Acknowledged
CAT-09	Missing Error Messages	Coding Style	Informational	Acknowledged
CAT-10	Improper Usage Of `public` And `external` Type	Gas Optimization	Informational	Acknowledged
CAT-11	Return Value Not Handled	Volatile Code	Informational	Acknowledged
CAT-12	Discussion For `takeFee`	Logical Issue	Informational	Acknowledged
CAT-13	Lack Of Input Validation	Volatile Code	Informational	Acknowledged

CAT-01 | Centralization Risks In Catcoin.sol

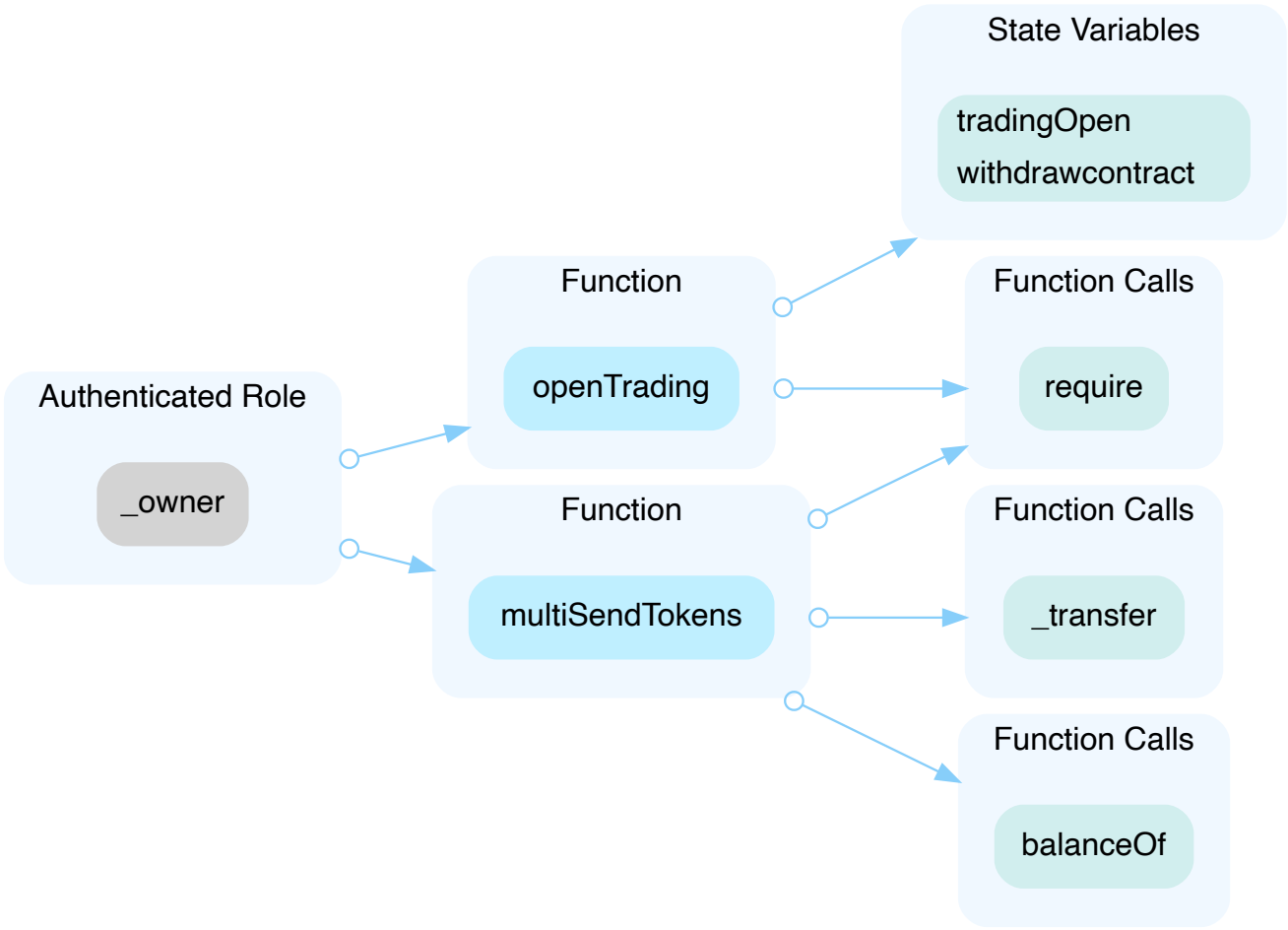
Category	Severity	Location	Status
Centralization / Privilege	**Major**	Catcoin.sol: 191, 195, 385, 441, 472, 479, 487, 494, 509, 519, 533, 538, 543, 576, 584, 723, 964	Resolved

Description

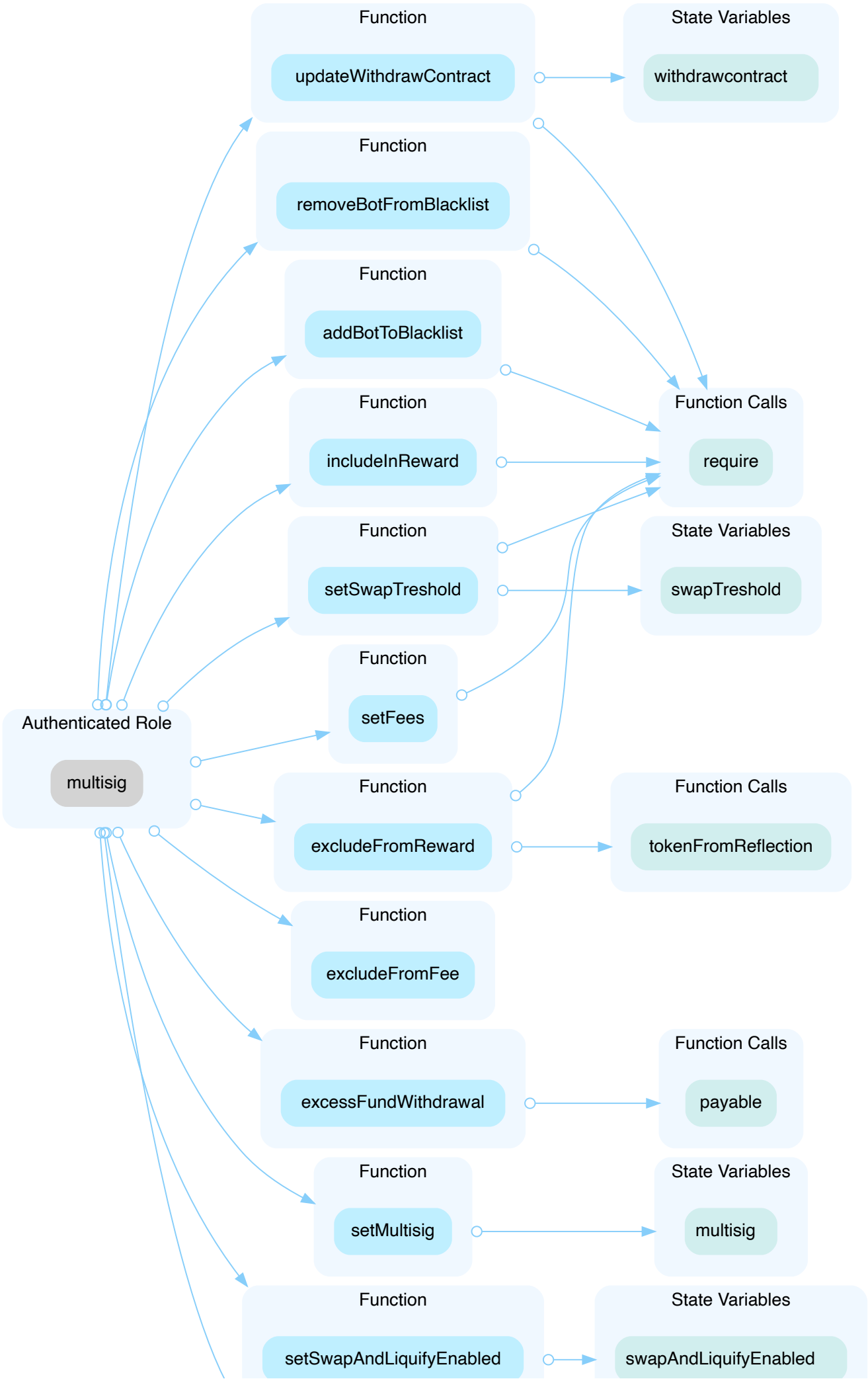
In the contract `Catcoin` the role `_devwallet` has authority over the functions shown in the diagram below. Any compromise to the `_devwallet` account may allow the hacker to take advantage of this authority.

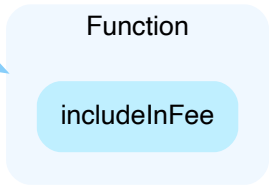


In the contract `Catcoin` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.

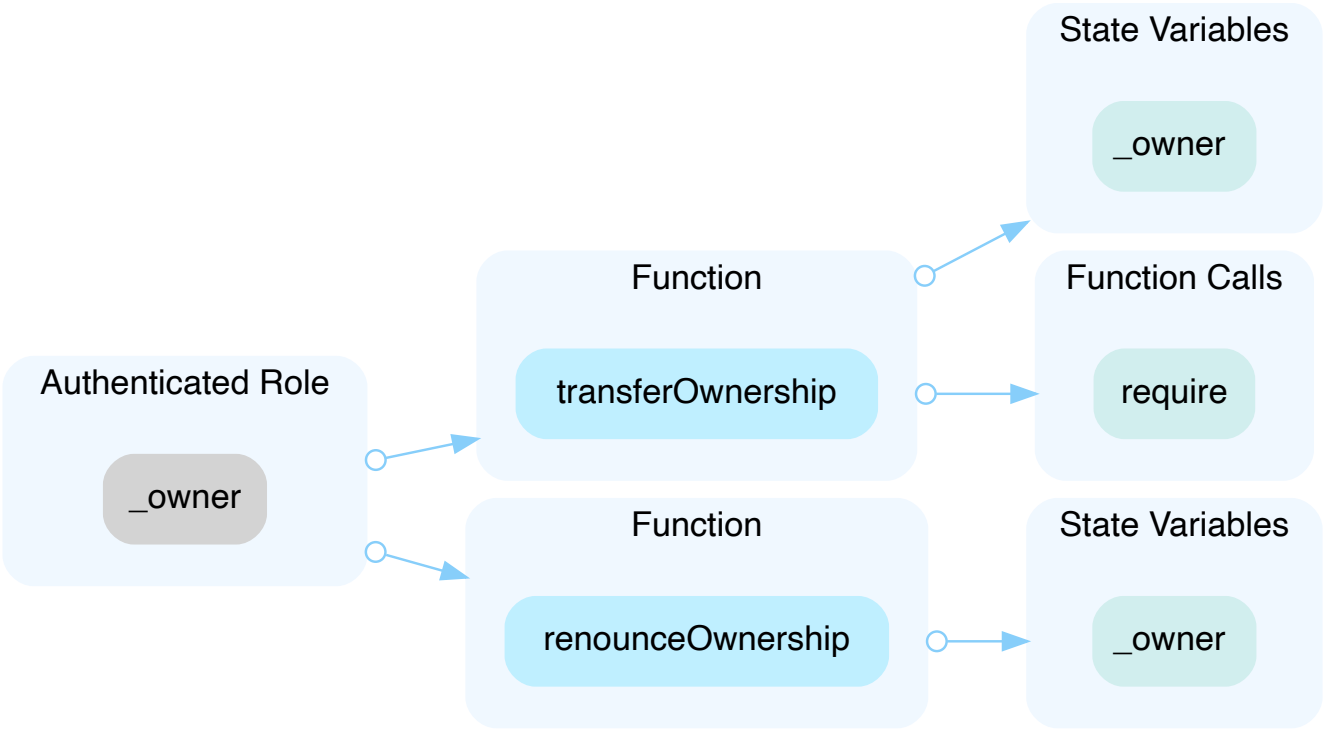


In the contract `Catcoin` the role `multisig`, which is a multi-signature contract with 4 owners now, has authority over the functions shown in the diagram below. If the `multisig` contract is updated to an EOA account, any compromise to the `multisig` account may allow the hacker to take advantage of this authority.


Function
updateWithdrawContract
State Variables
withdrawcontract
Function
removeBotFromBlacklist
Function
addBotToBlacklist
Function
includeInReward
Function Calls
require
Function
setSwapTreshold
State Variables
swapTreshold
Function
setFees
Authenticated Role
multisig
Function
excludeFromReward
Function Calls
tokenFromReflection
Function
excludeFromFee
Function
excessFundWithdrawal
Function Calls
payable
Function
setMultisig
State Variables
multisig
Function
setSwapAndLiquifyEnabled
State Variables
swapAndLiquifyEnabled



In the contract `Ownable` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations; AND

- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

[CatCoin]: The team resolved this issue by renouncing the ownership here: Transaction 0x14676d9a0477373339f70c3ce69c45a11e1fecea69e2120155de85abbb187c6b | BscScan

CAT-02 | Centralized Risk In `addLiquidity`

Category	Severity	Location	Status
Centralization / Privilege	**Major**	Catcoin.sol: 838	Mitigated

Description

```
833         uniswapV2Router.addLiquidityETH{value: ethAmount}(
834             address(this),
835             tokenAmount,
836             0,
837             0,
838             withdrawcontract,
839             block.timestamp
840         );
```

The `addLiquidity` function calls the `uniswapV2Router.addLiquidityETH` function with the `to` address specified as `withdrawcontract` for acquiring the generated LP tokens from the `Catcoin-BNB` pool. As a result, over time the `withdrawcontract` address will accumulate a significant portion of LP tokens. If the `withdrawcontract` is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences for the project as a whole.

Recommendation

We advise the `to` address of the `uniswapV2Router.addLiquidityETH` function call to be replaced by the contract itself, i.e. `address(this)`, and to restrict the management of the LP tokens within the scope of the contract's business logic. This will also protect the LP tokens from being stolen if the `withdrawcontract` account is compromised. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or via smart-contract-based accounts with enhanced security practices, f.e. Multisignature wallets.

Indicatively, here are some feasible solutions that would also mitigate the potential risk:

- Time-lock with reasonable latency, i.e. 48 hours, for awareness of privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO / governance / voting module to increase transparency and user involvement.

Alleviation

[CatCoin]: The `withdrawcontract` account is controlled by a multi-signature wallet, the wallet is the multi-signature wallet can be found here: Contract Address 0x49e231b76626bcd1c0d20706571a3eaafef52b22 | BscScan, and the wallet requires 3/5.

CAT-03 | Centralization Related Risks

Category	Severity	Location	Status
Centralization / Privilege	**Major**	Catcoin.sol: 579, 801, 805	Mitigated

Description

Over time, these accounts `withdrawcontract` and `_devwallet` will accumulate a significant amount of BNB.

```
579             payable(withdrawcontract).transfer(amountBNB);
```

```
800         if (marketingAmt > 0) {
801             payable(withdrawcontract).transfer(marketingAmt);
802         }
803
804         if (devAmt > 0) {
805             payable(_devwallet).transfer(devAmt);
806         }
```

Recommendation

In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or via smart-contract based accounts with enhanced security practices, f.e. Multisignature wallets.

Alleviation

[CatCoin]: The `withdrawcontract` account is controlled by a multi-signature wallet, the wallet is the multi-signature wallet can be found here: Contract Address 0x49e231b76626bcd1c0d20706571a3eaafef52b22 | BscScan, and the wallet requires 3/5 signatures.

The developer wallet has been remedied with including a similar fashion to multisig withdrawer and multisig controlled and distributed. Contract Address 0x2B6BB93FeE208e5A092F9258e9Da874F66F1d65D | BscScan Multisig: Contract Address 0xb4733eCF7337ecaefF1E031137c13c9217EE5743 | BscScan and this multisig requires 2/3 signatures.

CAT-04 | Initial Token Distribution

Category	Severity	Location	Status
Centralization / Privilege	**Major**	Catcoin.sol: 344	Acknowledged

Description

All of the Catcoin tokens are sent to the contract deployer when deploying the contract. This could be a centralization risk as the deployer can distribute Catcoin tokens without obtaining the consensus of the community.

Recommendation

We recommend the team to be transparent regarding the initial token distribution process, and the team shall make enough efforts to restrict the access of the private key.

Alleviation

[CatCoin]: The distribution was known to public and included with a file sharing addresses, the tokens were also distributed utilizing the contract with events to label sending.

CAT-05 | Missing Zero Address Validation

Category	Severity	Location	Status
Volatile Code	Minor	Catcoin.sol: 386, 734	Acknowledged

Description

Addresses should be checked before assignment or external call to make sure they are not zero addresses.

File: Catcoin.sol (Line 386, Function `Catcoin.setMultisig`)

```
        multisig = _multisig;
```

- `_multisig` is not zero-checked before being used.

File: Catcoin.sol (Line 734, Function `Catcoin.openTrading`)

```
        withdrawcontract = _withdrawcontract;
```

- `_withdrawcontract` is not zero-checked before being used.

Recommendation

We advise adding a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

[CatCoin]: The team acknowledged this issue and decided not to change the codebase. The `multisig` address can be changed to zero if the project no longer exists and is migrated in the future, or if multi-signature control of functions is renounced.

CAT-06 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Minor	Catcoin.sol: 236	Acknowledged

Description

The contract is serving as the underlying entity to interact with third-party PancakeSwap protocols. The scope of the audit treats 3rd party entities as black boxes and assumes their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Recommendation

We understand that the business logic of Catcoin requires interaction with the PancakeSwap protocol for adding liquidity to Catcoin-BNB pool and swapping tokens. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

[CatCoin]: The team acknowledged this issue and will constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

CAT-07 | Potential Gas Waste In `_transfer`

Category	Severity	Location	Status
Gas Optimization	Minor	Catcoin.sol: 748	Acknowledged

Description

After the contract gets fees from the previous transfer, the function `swapAndLiquify` will likely be called in the next transfer because `contractTokenBalance > 0` is true. This will result in too frequent swaps and the value of the transferred tokens may be lower than the gas cost.

Recommendation

We recommend using an appropriate value instead of zero as the preconditions of the swap.

Alleviation

[CatCoin]: The team acknowledged this issue and decided not to change the codebase this time.

CAT-08 | Useless Statement

Category	Severity	Location	Status
Logical Issue	Minor	Catcoin.sol: 795~798	Acknowledged

Description

The condition `unitBalance * 2 * (buyFee.dev + sellFee.dev) > address(this).balance` is useless, because the `marketingAmt` BNB has not been transferred.

Therefore, the check should be done after the transfer to `withdrawcontract` address happened.

Recommendation

Consider relocating the check as shown below:

```
if (marketingAmt > 0) {
    payable(withdrawcontract).transfer(marketingAmt);
}
uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
    address(this).balance
    ? address(this).balance
    : unitBalance * 2 * (buyFee.dev + sellFee.dev);

if (devAmt > 0) {
    payable(_devwallet).transfer(devAmt);
}
```

Alleviation

[CatCoin]: The team acknowledged this issue and decided not to change the codebase this time.

CAT-09 | Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	Catcoin.sol: 339, 442, 480, 554, 555, 861, 967	Acknowledged

Description

The **require** can be used to check for conditions and throw an exception if the condition is not met. It is better to provide a string message containing details about the error that will be passed back to the caller.

Recommendation

We advise adding error messages to the linked **require** statements.

Alleviation

[CatCoin]: The team acknowledged this issue and decided not to change the codebase this time.

CAT-10 | Improper Usage Of `public` And `external` Type

Category	Severity	Location	Status
Gas Optimization	Informational	Catcoin.sol: 393, 433, 437, 441, 447, 712	Acknowledged

Description

`public` functions that are never called by the contract could be declared as `external`. `external` functions are more efficient than `public` functions.

Recommendation

Consider using the external attribute for public functions that are never called within the contract.

Alleviation

[CatCoin]: The team acknowledged this issue and decided not to change the codebase this time.

CAT-11 | Return Value Not Handled

Category	Severity	Location	Status
Volatile Code	Informational	Catcoin.sol: 833~840	Acknowledged

Description

The return values of function `addLiquidityETH` are not properly handled.

```
833         uniswapV2Router.addLiquidityETH{value: ethAmount}(
834             address(this),
835             tokenAmount,
836             0,
837             0,
838             withdrawcontract,
839             block.timestamp
840         );
```

Recommendation

We recommend using variables to receive the return value of the functions mentioned above and handle both success and failure cases if needed by the business logic.

Alleviation

[CatCoin]: The team acknowledged this issue and decided not to change the codebase this time.

CAT-12 | Discussion For `takeFee`

Category	Severity	Location	Status
Logical Issue	Informational	Catcoin.sol: 844~852, 864	Acknowledged

Description

The variable `takeFee` indicates if the fee should be deducted when transferring tokens. But when `takeFee` is true and `sender` and `recipient` are also not equal to the `uniswapV2Pair`, the function `removeAllFee` will set all fee settings to zero. In line 864, after each call to the function `_tokenTransfer`, all fee settings will be set to zero when the function `removeAllFee` is called.

```
844         if (takeFee) {
845             removeAllFee();
846             if (sender == uniswapV2Pair) {
847                 setBuy();
848             }
849             if (recipient == uniswapV2Pair) {
850                 setSell();
851             }
852         }
853 
854         if (_isExcluded[sender] && !_isExcluded[recipient]) {
855             _transferFromExcluded(sender, recipient, amount);
856         } else if (!_isExcluded[sender] && _isExcluded[recipient]) {
857             _transferToExcluded(sender, recipient, amount);
858         } else if (_isExcluded[sender] && _isExcluded[recipient]) {
859             _transferBothExcluded(sender, recipient, amount);
860         } else {
861             require(!_isExcluded[sender] && !_isExcluded[recipient]);
862             _transferStandard(sender, recipient, amount);
863         }
864         removeAllFee();
```

At the end of the `_tokenTransfer` function as shown below, why not restore the fees but still remove all fees?

```
843     function _tokenTransfer(address sender, address recipient, uint256 amount, bool
takeFee) private {
844         if (takeFee) {
845             removeAllFee();
846             ...
847         }
848 
```

```
849         if (_isExcluded[sender] && !_isExcluded[recipient]) {
850             _transferFromExcluded(sender, recipient, amount);
851         }
852         ...
853         removeAllFee();
854     }
```

Recommendation

Consider checking whether the implementation match the design.

Alleviation

[CatCoin]: The team acknowledged this issue and will fix this issue in the future.

CAT-13 | Lack Of Input Validation

Category	Severity	Location	Status
Volatile Code	Informational	Catcoin.sol: 543~553, 768	Acknowledged

Description

Lack of input validation when setting fees in the `setFees` function to ensure the below denominator to be greater than zero.

```
        uint256 denominator = (buyFee.liquidity +
            sellFee.liquidity +
            buyFee.marketing +
            sellFee.marketing +
            buyFee.dev +
            sellFee.dev) * 2;
```

Recommendation

Consider checking the fees to ensure the denominator is greater than zero.

Alleviation

[CatCoin]: Values can be 0 in some cases and do not need to be checked.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings 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.

Logical Issue

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

Volatile Code

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

Coding Style

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

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

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 you ("Customer" or 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, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

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.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

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.