FUSION TECH

HUSKY

Security Assessment

TABLE OF CONTENTS

Summary

This report has been prepared to discover issues and vulnerabilities in the source code of the project as well as any contract dependencies that were not part of an officially recognized library.A comprehensive examination has been performed, utilizing Static Analysisand Manual Reviewtechniques. The auditingprocess pays specialattention to the following considerations:

Testing the smart contracts against both commonand uncommon attackvectors. Assessing the codebase to ensure compliancewith current best practices and industry standards.Ensuring contract logic meets the specifications and intentions of the client.
Cross referencing contract structure and implementation againstsimilar smart contracts produced by industryleaders.
Thorough line-by-line manual review of the entirecodebase by industryexperts.

The security assessment resulted in findings that ranged from critical to informational. We recommendaddressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could betterserve the projectfrom the securityperspective:

Enhance generalcoding 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, especiallycontracts that are verified in public;
Provide more transparency on privileged activities once the protocolis live.

Project Summary

Project Name	HUSKY
Platform	CORE DAO
Language	Solidity
Codebase	https://scan.coredao.org/address/0x704c2bFF5b79839103551f6dB8E89fD131Ac5569
Commit	87cc90787612fd4537ftf5b845nv543g7l10i472

Audit Summary

Delivery Date	Feb,25 2023
Audit Methodology	Static Analysis, Manual Review
Key Components	Core Token

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	2	0	0	2	0	2
Medium	1	0	0	2	0	1
Minor	0	0	0	3	0	0
Informational	3	0	0	6	0	3
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
CKP	contract.sol	fd5b505af11580156108c0573d6000803e3d6000fd5b505050506040513d6020

4 files audited • 3 files with Acknowledged findings • 1 file with Resolved findings

ID	Repo	Commit	File	SHA256 Checksum
APB	coredao-org/dao-contracts	1a550d5	contracts/AirdropPool.sol	6b107e9cb0aea38712d01b7e40a7ed18aeb91a0e2aa7dd1ccd880b266c0dc728
COR	coredao-org/dao-contracts	1a550d5	contracts/CORESales.sol	0323126fc721ae5ea737ad33aaeb2568a26b5eeb2fd011006f1cfce1dee7e648
TVB	coredao-org/dao-contracts	1a550d5	contracts/TeamVesting.sol	939147a2ac8de592227101e2fb9aeeb4c2d00eefac4679d96eab4f4eef78c635
SMB	coredao-org/dao-contracts	1a550d5	contracts/lib/SafeMath.sol	6eeb4a240710a44001fb885f5000495e673c553a0c19e878265a523ad66095ea

Overview

External Dependencies

Thecontract serves as the underlying entity to interactwith third-party protocols (token- wapping). The scopeof the audit treats third-party entities as blackboxes and assumestheir functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolenassets.

Privileged Functions

The contract contains the followingprivileged functions that are restricted by role with the modifier. They are used to modify the contractconfigurations and addressattributes. We grouped these functions below.

- event OperatorTransferred()
- event OwnershipTransferred()
- function allowance()
- function approve()
- function burn()
- function decreaseAllowance()
- function owner()
- function renounceOwnership()
- setSwapAndLiquifyEnabled()
- setRouterAddress()

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

01 | Centralization Risk in Function

Description

The addLiquidity()_hasLiqBeenAdded() function calls the coreV2Router.addLiquidityETH function with the to() address specified as owner() for acquiring the generated LP tokens from the corresponding pool. As a result, over time the _owner address will accumulate a significant portionof LP tokens. If _owner the is an EOA (Externally Owned Account),mishandling of its private key can have devastating consequences to the projectas a whole.

Recommendation

We advise to() the address of the coreV2Router.addLiquidityETH() function call to be replaced by the contract() itself, i.e. address(this) , and to restrict the managementof the LP tokens within the scope of the contract's businesslogic. This will also protectthe LP tokens from being stolen if the _owner() account is compromised. In general, we strongly recommend centralized privileges or roles in the protocolto be improved via a decentralized mechanism or via smart-contract based accounts with enhanced securitypractices, f.e.Multisignature wallets().

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

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

02 | Centralization Risk in Contract

Category	Severity	Location	Status
Logical Issue	Medium	projects/contract.sol (98ba012): 243	Acknowledged

Description

In the contract CoinTokens(), the role _owner() has the authority over the following function:

- excludeFromReward() / includeInReward() ; the owner of the contractcan exclude/include an account from/inrewards.
- excludeFromFee() / includeInFee() : the owner of the contractcan exclude/include an account from/in fee.
- setTaxFeePercent() : the owner of the contractcan set the percentage of the tax fee.
- setDevFeePercent() : the owner of the contract can set the percentageof the dev fee.
- setLiquidityFeePercent() : the owner of the contract can set the percentageof liquidity fee.
- setMaxTxPercent() : the owner of the contract can set the maximum transaction amount.
- setDevWalletAddress() : the owner of the contractcan update the arbitrary address.
- setRouterAddress() : the owner of the contractcan set any arbitrary addressas the router address.
- setNumTokensSellToAddToLiquidity() : the owner of the contract can set the thresholdto trigger liquidity-adding process.

Any compromise to the _owner() account may allow the hacker to take advantageof this and modify the significant state of the contract,thus introducing centralization risk.

03 | APPROACH & METHODS

Description

This report has been prepared for HUSKY to discover issues and vulnerabilities in the source code of the HUSKY project as well as any contract dependencies that were not part of an officially recognized library Acomprehensive examination has been performed utilizing Manual Review and Static Analysis techniques

- 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 byindustry leader
- 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 addressingthese findings to ensure a high level of security standards and industry practices We suggest recommendationsthat could better serve the project from the security perspective

- testing the smart contracts against both common and uncommon attack vectors
- 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

04 | SetAllowance()

Description

setAllowance() currently poses a risk of a race condition Consider the scenario

- operator() is allowed to applyFunds() up to 1000
- dao wants to increase the allowance by 100 and calls setAllowance(1100)
- However before this transaction was executed operator calls applyFunds(1000)
- After the allowance is increased operator calls applyFunds(1100)
- In total operator has spent which was not expected by dao

Recommendation

To prevent a possible race condition we recommend introducing increaseAllowance() and decreaseAllowance()

05 | Initial Token Distribution

Category	Severity	Location	Status
Logical Issue	Medium	projects/contract.sol (98ba012): 817	Acknowledged

Description

All of the tokens are sent to the contract deployer when deploying the contract. This could be a centralization risk as the deployer can distribute those tokens withoutobtaining the consensusof the community.

Recommendation

We recommend the team to be transparent regarding the initialtoken distribution process.

06 | Lack of Return Value Handling

Category	Severity	Location	Status
Volatile Code	Minor	projects/contract.sol (98ba012): 843	Acknowledged

Description

The return valuesof function tryADD() are properly handled.

```
function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
    unchecked {
        uint256 c = a + b;
        if (c < a) return (false, 0);
        return (true, c);
function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
    unchecked {
        if (b > a) return (false, 0);
        return (true, a - b);
```

Recommendation

We recommend using variables to receive the return value of the functions mentionedabove and handleboth success and failure cases if neededby the business logic.

07 | UNLOCKED COMPILER VERSION

Language Specific	Informational	contracts/AirdropPool.sol (base): 1; contracts/CORESales.sol (base): 1; contracts/TeamVesting.sol (base): 1; contracts/lib/SafeMath.sol (base): 1	Resolved

Description

The contracts cited have an unlocked compiler version An unlocked compiler version in the source code of thecontract permits the user to compile it at or above a particular version This in turn leads to differences in thegenerated bytecode between compilations due to differing compiler version numbers This can lead to an ambiguitywhen debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span ofmultiple compiler versions rather than a specific one

Recommendation

We recommend the compiler version is instead locked at the lowest version possible that the contract can becompiled at For example for version v0.8.0 the contract should contain the following line

```
pragma solidity ^0.8.0;
```

08 | Lack of Error Message

Category	Severity	Location	Status
Coding Style	Informational	projects/contract.sol (98ba012): 560	Acknowledged

Description

The require statement can be used to check for conditions and throw an exception if the condition is not met.It is better to providea string messagecontaining details about the errorthat will be passed back to the caller.

Recommendation

We advise refactoring the linked codes as below:

```
function mod(uint256 a, uint256 b) internal pure returns (uint256) {
    return a % b;
}
```

09 | Redundant Code

Category	Severity	Location	Status
Logical Issue	Informational	projects/contract.sol (98ba012): 862	Acknowledged

Description

The condition! _isExcluded[sender] & !_isExcluded[recipient] can be included in else .

Recommendation

The following code can be removed:

```
... else if (!_isExcluded[sender] && !_isExcluded[recipient]) {
    _transferStandard(sender, recipient, amount);
} ...
```

10 | Typos In The Contract

Category	Severity	Location	Status
Coding Style	Informational	projects/contract.sol (98ba012): 470, 670	Acknowledged

Description

There are several typos in the code and comments.

1. In the following code snippet, tokensIntoLiquidity() should be tokensIntoLiquidity()

```
function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
    unchecked {
        // Gas optimization: this is cheaper than requiring 'a' not being zero, but the
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522
        if (a == 0) return (true, 0);
        uint256 c = a * b;
        if (c / a != b) return (false, 0);
        return (true, c);
```

2. recieve() should be recieve() _swapping() should be _swapping() in the line of comment //to _recieve ETH from coreV2Router when swaping() .

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 mechanismto relocate funds.

Logical Issue

Logical Issue findingsdetail a faultin the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Volatile Code

Volatile Code findingsrefer to segmentsof code that behave unexpectedly on certain edge cases that may resultin a vulnerability.

Coding Style

Coding Style findingsusually do not affect the generated byte-code but rather commenton how to make the codebase more legible and, as a result, easilymaintainable.

Inconsistency

Inconsistency findings referto functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing differentrequire statements on the input variables than a setterfunction.

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 specifiedcommit.

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

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimerand limitation of liability) set forth in the ServicesAgreement, or the scope of services, and terms and conditions providedto you ("Customer" or the "Company") in connection with the Agreement. This report providedin connection with the Serviceset forth in the Agreementshall be used by the Company only to the extent permittedunder 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 copiesbe delivered to any otherperson other than the Company,without FusionTech prior written consentin 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 FusionTech 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, businessmodel 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 shouldbe 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. FusionTech's position is that each companyand individual are responsible for their own due diligenceand continuous security.
FusionTech's goal is to help reduce the attack vectorsand 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 FusionTech is subject to dependencies and under continuing development. You agree that your access and/or use, includingbut 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 tokensare 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 PRODUCTSOR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUTWARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, FusionTech HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, FusionTech SPECIFICALLY DISCLAIMS ALL IMPLIEDWARRANTIES 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, FusionTech 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 HARMFULCODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, FusionTech 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 STANDARDSOR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHERFusionTech NOR ANY OF FusionTech'SAGENTS 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. FusionTech 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, RESULTINGFROM 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 THETHIRD-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 PURPOSENOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIESBE DELIVERED TO, ANY OTHERPERSON WITHOUT FusionTech'SPRIOR WRITTEN CONSENTIN EACH INSTANCE.

NOTHIRD PARTY OR ANYONE ACTINGON BEHALF OF ANY THEREOF,SHALL BE A THIRD PARTYOR OTHER BENEFICIARY OF SUCH SERVICES,ASSESSMENT REPORT, AND ANY ACCOMPANYING

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

About

Founded in 2022 by leading academics in the field of Computer Science, FusionTech is going to be a leading blockchain security company that serves to verify the security and correctness of smart contracts KYC 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 ourclients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

FUSION TECH