ASSURE DEFI®
THE VERIFICATION GOLD STANDARD

Security Assessment:

Munch TOKEN

July 17, 2024

- Audit Status: **Fail**
- Audit Edition: **Advance**

VERIFIED BY ASSURE DEFI ®
INTEGRITY ★ TRUST ★ CREDIBILITY

Risk Analysis

Classifications of Manual Risk Results

Classification	Description
Critical	Danger or Potential Problems.
High	Be Careful or Fail test.
Medium	Pass, Not-Detected or Safe Item.
Low	Function Detected

Manual Code Review Risk Results

Contract Privilege	Description
Buy Tax	3%
Sale Tax	3%
Cannot Buy	Pass
Cannot Sale	Pass
Max Tax	3%
Modify Tax	Yes
Fee Check	Pass
Is Honeypot?	Not Detected
Trading Cooldown	Not Detected
Can Pause Trade?	Pass
Pause Transfer?	Not-Detected
Max Tx?	Fail
Is Anti Whale?	Detected
Is Anti Bot?	Not-Detected

Contract Privilege	Description
Is Blacklist?	Not-Detected
Blacklist Check	Pass
is Whitelist?	Detected
Can Mint?	Pass
Is Proxy?	Not Detected
Can Take Ownership?	Not Detected
Hidden Owner?	Not-Detected
Owner	0x
Self Destruct?	Not Detected
External Call?	Not-Detected
Other?	Not Detected
Holders	0
Auditor Confidence	Medium
KYC Present	No
KYC URL	

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

Project Overview

Token Summary

Parameter	Result
Address	0x30683d46edD7E2A52402e5301B14dB33BD4Ff550
Name	Munch
Token Tracker	Munch (MUNCH)
Decimals	18
Supply	1,000,000,000
Platform	
compiler	0.8.19
Contract Name	MunchToken
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	
Payment Tx	Corporate

Main Contract Assessed Contract Name

Name	Contract	Live
Munch	0x30683d46edD7E2A52402e5301B14dB33BD4Ff550	Yes

TestNet Contract Assessed Contract Name

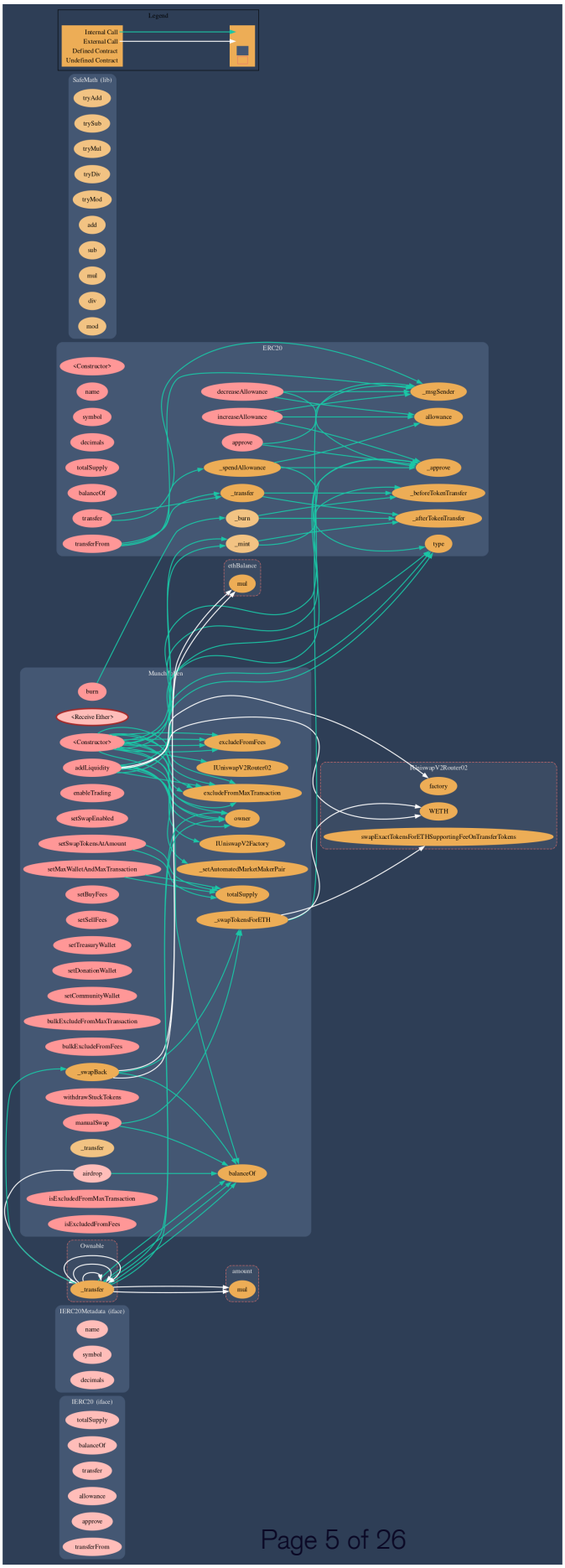
Name	Contract	Live
Munch	0xE44d49E61BA9Ee132BdB4035145Bc18cE2FE19f3	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
Munch	87979b8d1e50ad0bdd71e43ea5f2ea592a6eb4ac	MunchToken.sol
Munch		.sol
Munch		.sol
Munch		.sol
Munch		.sol
Munch		.sol

Call Graph

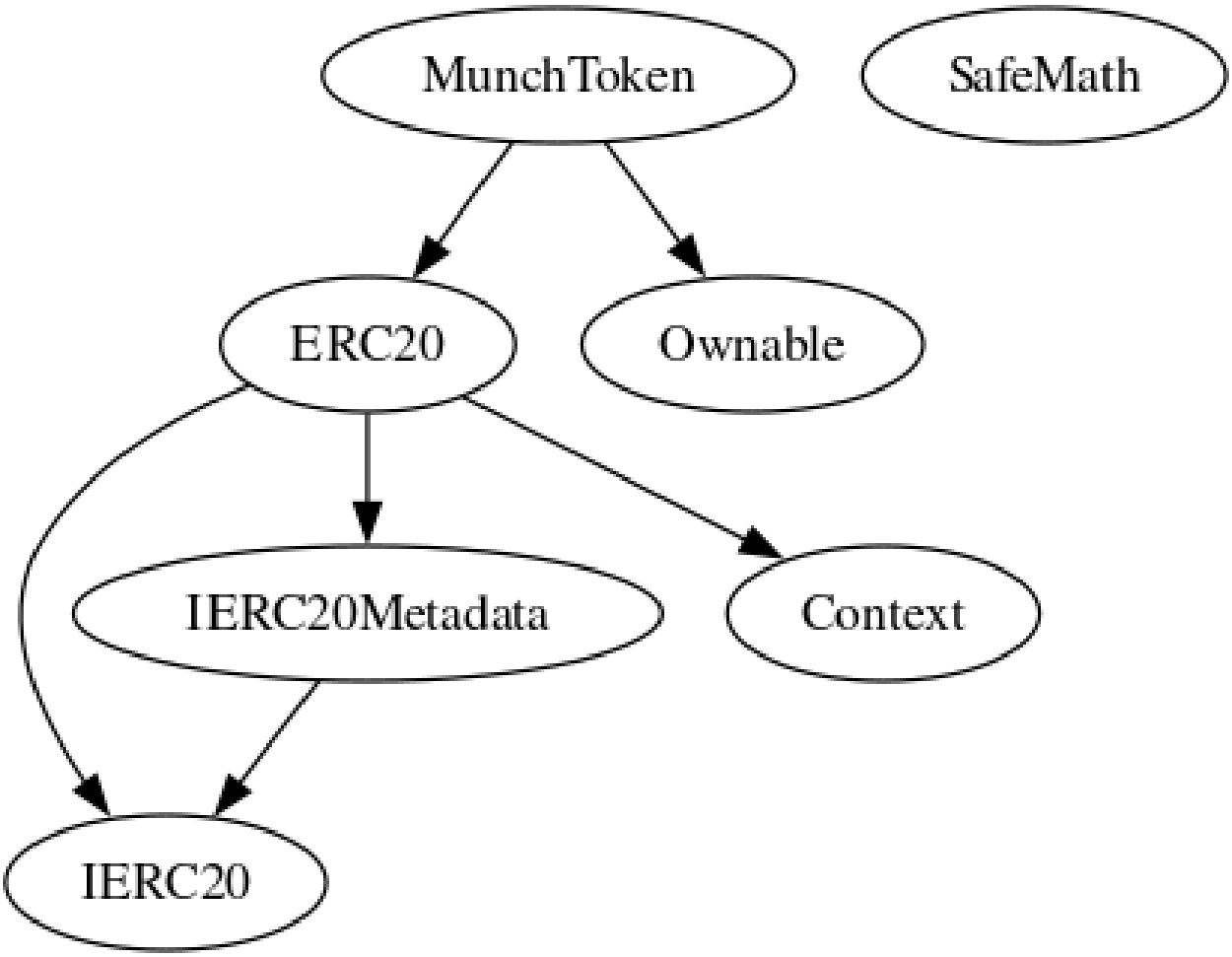
The contract for Munch has the following call graph structure.



Inheritance

The contract for Munch has the following inheritance structure.

The Project has a Total Supply of 1,000,000,000



Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

Function Name	Parameters	Visibility
addLiquidity		Public
enableTrading		Public
setSwapEnabled	bool value	Public
setSwapTokensAtAmount	uint256 amount	Public
setMaxWalletAndMaxTransaction	uint256 _maxTransaction, uint256 _maxWallet	Public
setBuyFees	uint256 _treasuryFee, uint256 _donationFee, uint256 _communityFee	Public
setSellFees	uint256 _treasuryFee, uint256 _donationFee, uint256 _communityFee	Public
setTreasuryWallet	address _treasuryWallet	Public
setDonationWallet	address _donationWallet	Public

Function Name	Parameters	Visibility
setCommunityWallet	address _communityWallet	Public
excludeFromMaxTransaction	address account, bool value	Public
bulkExcludeFromMaxTransaction	address[] calldata accounts, bool value	Public
excludeFromFees	address account, bool value	Public
bulkExcludeFromFees	address[] calldata accounts, bool value	Public
manualSwap		Public
withdrawStuckTokens	address tkn	Public
airdrop	address[] calldata addresses, uint256[] calldata tokenAmounts	External

MUNCH-01 | Potential Sandwich Attacks.

Category	Severity	Location	Status
Security	Low	MunchToken.sol: L: 1198 C: 14	Detected

Description

A sandwich attack might happen when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by frontrunning (before the transaction being attacked) a transaction to purchase one of the assets and make profits by back running (after the transaction being attacked) a transaction to sell the asset. The following functions are called without setting restrictions on slippage or minimum output amount, so transactions triggering these functions are vulnerable to sandwich attacks, especially when the input amount is large:

- swapExactTokensForETHSupportingFeeOnTransferTokens()
- addLiquidityETH()

Remediation

We recommend setting reasonable minimum output amounts, instead of 0, based on token prices when calling the aforementioned functions.

Referrences:

What Are Sandwich Attacks in DeFi — and How Can You Avoid Them?.

MUNCH-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	Informational	MunchToken.sol: L: 834 C: 14,L: 839 C: 14,L: 865 C: 14,L: 871 C: 14,L: 875 C: 14,L: 887 C: 14,L: 887 C: 14,L: 903 C: 14,L: 918 C: 14,L: 934 C: 14,L: 941 C: 14,L: 948 C: 14,L: 955 C: 14,L: 963 C: 14,L: 973 C: 14,L: 978 C: 14,L: 988 C: 14,L: 992 C: 14	Detected

Description

The following functions are declared as public and are not invoked in any of the contracts contained within the projects scope:

Function Name	Parameters	Visibility
burn	uint256 amount	Public
addLiquidity		Public
enableTrading		Public
setSwapEnabled	bool value	Public
setSwapTokensAtAmount	uint256 amount	Public
setMaxWalletAndMaxTransaction	uint256 _maxTransaction, uint256 _maxWallet	Public
setBuyFees	uint256 _treasuryFee, uint256 _donationFee, uint256 _communityFee	Public
setSellFees	uint256 _treasuryFee, uint256 _donationFee, uint256 _communityFee	Public
setTreasuryWallet	address _treasuryWallet	Public
setDonationWallet	address _donationWallet	Public

Function Name	Parameters	Visibility
setCommunityWallet	address _communityWallet	Public
excludeFromMaxTransaction	address account, bool value	Public
bulkExcludeFromMaxTransaction	address[] calldata accounts, bool value	Public
excludeFromFees	address account, bool value	Public
bulkExcludeFromFees	address[] calldata accounts, bool value	Public
manualSwap		Public
withdrawStuckTokens	address tkn	Public
airdrop	address[] calldata addresses, uint256[] calldata tokenAmounts	External
_setAutomatedMarketMakerPair	address pair, bool value	Internal
_swapBack		Internal
_swapTokensForETH	uint256 tokenAmount	Internal

The functions that are never called internally within the contract should have external visibility

Remediation

We advise that the function's visibility specifiers are set to external, and the array-based arguments change their data location from memory to calldata, optimizing the gas cost of the function.

References:

external vs public best practices.

MUNCH-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	Low	MunchToken.sol: L: 973 C: 14,L: 955 C: 14,L: 955 C: 14	Not-Detected

Description

The given input is missing the check for the non-zero address.

The given input is missing the check for the onlyOwners need to be revisited for require..

Remediation

We advise the client to add the check for the passed-in values to prevent unexpected errors as below:

```
...
 require(receiver != address(0), "Receiver is the zero address");
...
...
require(value X limitation, "Your not able to do this function");
...
```

We also recommend customer to review the following function that is missing a required validation. onlyOwners need to be revisited for require..

MUNCH-04 | Centralized Risk In addLiquidity.

Category	Severity	Location	Status
Coding Style	High	MunchToken.sol: L: 860 C: 14	Detected

Description

uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this), tokenAmount, 0, 0, owner(), block.timestamp);

The addLiquidity function calls the uniswapV2Router.addLiquidityETH function with the to address specified as owner() for acquiring the generated LP tokens from the MUNCH-WBNB pool.

As a result, over time the _owner address will accumulate a significant portion of LP tokens.If the _owner is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences to the project as a whole.

Remediation

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 _owner 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.

1. Indicatively, here are some feasible solutions that would also mitigate the potential risk:
2. Time-lock with reasonable latency, i.e. 48 hours, for awareness on privileged operations;
3. Assignment of privileged roles to multi-signature wallets to prevent single point of failure due to the private key;

Introduction of a DAO / governance / voting module to increase transparency and user involvement

Project Action

MUNCH-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	Low	MunchToken.sol: L: 918 C: 14,L: 865 C: 14,L: 871 C: 14,L: 875 C: 14,L: 887 C: 14,L: 903 C: 14,L: 918 C: 14	Detected

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.The linked code does not create an event for the transfer.

Remediation

Emit an event for critical parameter changes. It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

MUNCH-07 | State Variables could be Declared Constant.

Category	Severity	Location	Status
Coding Style	Low	MunchToken.sol: L:720	Not Detected

Description

Constant state variables should be declared constant to save gas.

```
IUniswapV2Router02
```

Remediation

Add the constant attribute to state variables that never changes.

https://docs.soliditylang.org/en/latest/contracts.html#constant-state-variables

MUNCH-08 | Dead Code Elimination.

Category	Severity	Location	Status
Coding Style	Low	MunchToken.sol: L: 4, L: 809, C:14	Detected

Description

Functions that are not used in the contract, and make the code s size bigger.

```
ABIEncoderV2
_previousFee
```

Remediation

Remove unused functions. dead-code elimination (also known as DCE, dead-code removal, dead-code stripping, or dead-code strip) is a compiler optimization to remove code which does not affect the program results. Removing such code has several benefits: it shrinks program size, an important consideration in some contexts, and it allows the running program to avoid executing irrelevant operations, which reduces its running time. It can also enable further optimizations by simplifying program structure.

https://docs.soliditylang.org/en/latest/cheatsheet.html

MUNCH-14 | Unnecessary Use Of SafeMath

Category	Severity	Location	Status
Logical Issue	Medium	MunchToken.sol: L: 488 C: 0	Detected

Description

The SafeMath library is used unnecessarily. With Solidity compiler versions 0.8.0 or newer, arithmetic operations
will automatically revert in case of integer overflow or underflow.
library SafeMath {
An implementation of SafeMath library is found.
using SafeMath for uint256;
SafeMath library is used for uint256 type in contract.

Remediation

We advise removing the usage of SafeMath library and using the built-in arithmetic operations provided by the
Solidity programming language

Project Action

MUNCH-18 | Stop Transactions by using Enable Trade.

Category	Severity	Location	Status
Logical Issue	Critical	MunchToken.sol: L: 865 C: 14	Detected

Description

Enable Trade is presend on the following contract and when combined with Exclude from fees it can be considered a whitelist process, this will allow anyone to trade before others and can represent and issue for the holders.

Remediation

We recommend the project owner to carefully review this function and avoid problems when performing both actions.

Project Action

MUNCH-20 | Potential Reentrancy in _swapTokensForETH.

Category	Severity	Location	Status
Coding Best Practices	Medium	MunchToken.sol: L: 1190	Detected

Description

The function uses a call to transfer ETH which can be exploited for reentrancy.

Remediation

Implement reentrancy guard or use checks-effects-interactions pattern.

Project Action

MUNCH-21 | Lack of Emergency Withdraw.

Category	Severity	Location	Status
Logical Issue	Medium	MunchToken.sol:	Detected

Description

No function to withdraw stuck tokens or ETH.

Remediation

Implement an emergency withdraw function.

Project Action

Technical Findings Summary

Classification of Risk

Severity	Description
Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
High	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
Low	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

Findings

Severity	Found	Pending	Resolved
Critical	1	1	0
High	1	1	0
Medium	3	3	0
Low	5	5	0
Informational	1	1	0
Total	11	11	0

Social Media Checks

Social Media	URL	Result
Twitter	https://x.com/munchtoken	Pass
Other		N/A
Website	https://munchproject.io	Pass
Telegram	https://t.me/MUNCHProjectportal	Pass

We recommend to have 3 or more social media sources including a completed working websites.

Social Media Information Notes:

Auditor Notes: undefined

Project Owner Notes:

Audit Result

Final Audit Score

Review	Score
Security Score	70
Auditor Score	75

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project most pass and provide all the data needed for the assessment. Our Passing Score has been changed to 85 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below.

Audit Fail

Assessment Results

Important Notes:

- Overall Classification:ı
- Performance: Mediumı
- Score: 70/100ı
- The MunchToken contract has several useful features but also presents significant risks, primarily due to centralization and complex fee mechanisms. Key areas for improvement include, optimizing gas usage, adding emergency withdraw functions, and securing against reentrancy attacks. The contract does not pass the audit with a score of 70, below the passing score of 85. Addressing the unresolved issues is essential for enhancing the contract's security and robustness.

Auditor Score =75
Audit Fail

Appendix

Finding Categories

Centralization / Privilege

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

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM 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 howblock.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing 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 mayresult in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to makethe 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 setterfunction.

Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

Disclaimer

Assure Defi has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocation for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and Assure Defi is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will Assure Defi or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by Assure Defi are subject to dependencies and are under continuing development. You agree that your access 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 with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.

ASSURE DEFI™
THE VERIFICATION GOLD STANDARD