

Vega Vegatoken Audit

- [1 Summary](#)
- [2 Audit Scope](#)
- [3 System Overview](#)
- [4 Key Observations/Recommendations](#)
- [5 Security Specification](#)
 - [5.1 Actors](#)
 - [5.2 Trust Model](#)
 - [5.3 Important Security Properties](#)
- [6 Issues](#)
 - [6.1 ERC20Lockable - inconsistent locking status](#) Minor ✓ Fixed
- [7 Tool-Based Analysis](#)
 - [7.1 MythX](#)
 - [7.2 Ethlint](#)
 - [7.3 Surya](#)
- [Appendix 1 - Disclosure](#)

Date	January 2020
Lead Auditor	Martin Ortner
Co-auditors	Gonçalo Sá

1 Summary

ConsenSys Diligence conducted a security audit on the Vega token, an ERC20 compliant token for the [Vega protocol](#), a protocol for creating and trading derivatives on a fully decentralized network.

Audit Changelog	Date
Initial report delivery.	2020-01-29
Diff highlighting fixes of the issues raised in the report was provided by the Vega team.	2020-02-10

Audit Changelog	Date
Fixes acknowledged by the audit team and final report delivery.	2020-02-12

2 Audit Scope

This audit covered the following files:

Source repository commit# : [vega-protocol/vega_token.git#672d747e2f38cd9f51ee7b4cd99ceb02a3573bfb](https://github.com/vega-protocol/vega_token.git#672d747e2f38cd9f51ee7b4cd99ceb02a3573bfb)

File Name	SHA-1 Hash
contracts/Vega-Token.sol	b92b3c54b2f47a88fa9e84534046b462dbae9aa
contracts/Address.sol	1213b0f150dd5e3f694c3721c44cb5cc3202b743
contracts/ERC20Detailed.sol	7e4d00c462120565201f28361b29201d1bfe0a34
contracts/IERC20.sol	72c15b6a16b7dc92e69ff97ccfe1958d9948e200
contracts/ERC20Lockable.sol	377447995444beee2b3c5342bfa9b1bbc1d08356
contracts/SafeMath.sol	c8bda5eb19c16d34bc48bf115229a9b967feb6ef
contracts/ERC20StaticSupply.sol	bf3e66af74470eed08d0e0f82b9a98705a745c7c
contracts/Ownable.sol	12ec51ec8a3b4eed6326434fd0f5926b40602778
contracts/Roles.sol	2c85acf184ae36f96ebafd8f6e26232ea459a711
contracts/SafeERC20.sol	ebd65ea9a0cdbc29bbbf651a1076d51be031443

- **contracts/Roles.sol**
 - is an unmodified copy of [OpenZeppelin/contracts/access/Roles.sol#67bca85](https://github.com/OpenZeppelin/contracts/access/Roles.sol#67bca85).
 - is the latest version available to date.
 - is **not referenced** throughout the codebase.
- **contracts/Address.sol**
 - is an unmodified copy of [OpenZeppelin/contracts/utils/Address.sol#18d7e24](https://github.com/OpenZeppelin/contracts/utils/Address.sol#18d7e24).
 - is **not** the latest version available to date.
- **contracts/SafeERC20.sol**
 - is a copy of [OpenZeppelin/contracts/token/ERC20/SafeERC20.sol#5d34dbe](https://github.com/OpenZeppelin/contracts/token/ERC20/SafeERC20.sol#5d34dbe) with minor modifications (adjusted import pragma).

- is **not referenced** throughout the codebase
- **contracts/Ownable.sol**
 - is a copy of [OpenZeppelin/contracts/ownership/Ownable.sol#f095b62](#) with no logic changing modifications to the contract.
- **contracts/SafeMath.sol**
 - is a copy of [OpenZeppelin/contracts/math/SafeMath.sol#c4bb7b7](#) with no logic changing modifications to the contract.
- **contracts/IERC20.sol, contracts/ERC20Detailed.sol**
 - are a unmodified copies of
 - [OpenZeppelin/contracts/token/ERC20/IERC20.sol#7552af9](#) and
 - [OpenZeppelin/contractstoken/ERC20/ERC20Detailed.sol#7552af9](#)

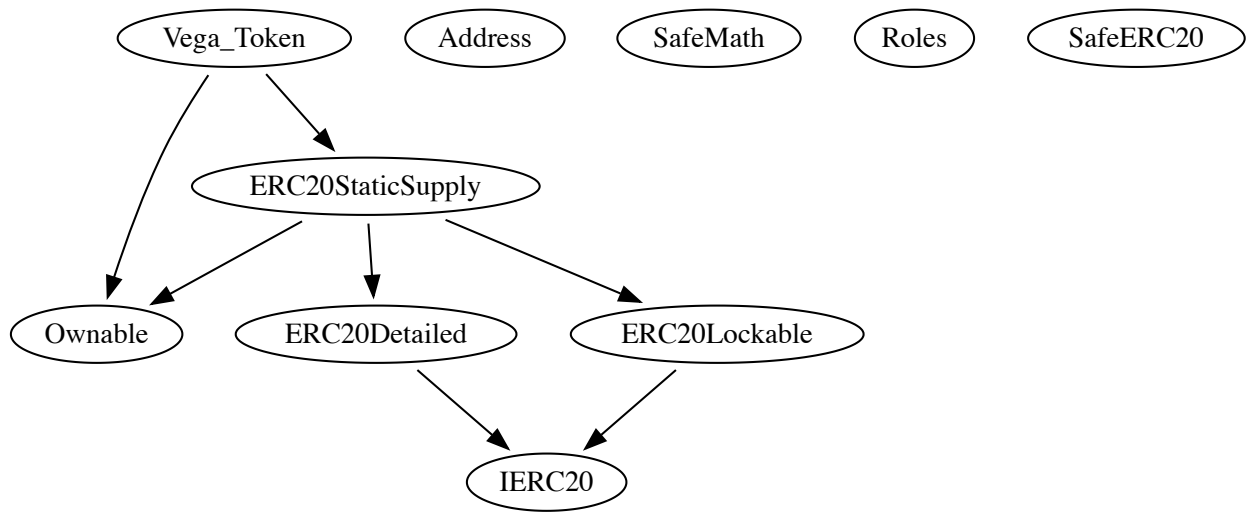
The audit activities can be grouped into the following three broad categories:

1. **Security:** Identifying security related issues within the contract.
2. **Architecture:** Evaluating the system architecture through the lens of established smart contract best practices.
3. **Code quality:** A full review of the contract source code. The primary areas of focus include:
 - Correctness
 - Readability
 - Scalability
 - Code complexity
 - Quality of test coverage

3 System Overview

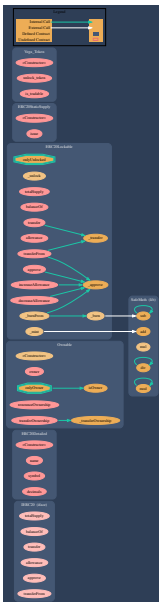
This section describes the top-level contracts, their inheritance structure, actors, permissions and contract interactions. Please refer to [section 5 - Security Specification](#) for a security-centric view on the system.

Inheritance Structure



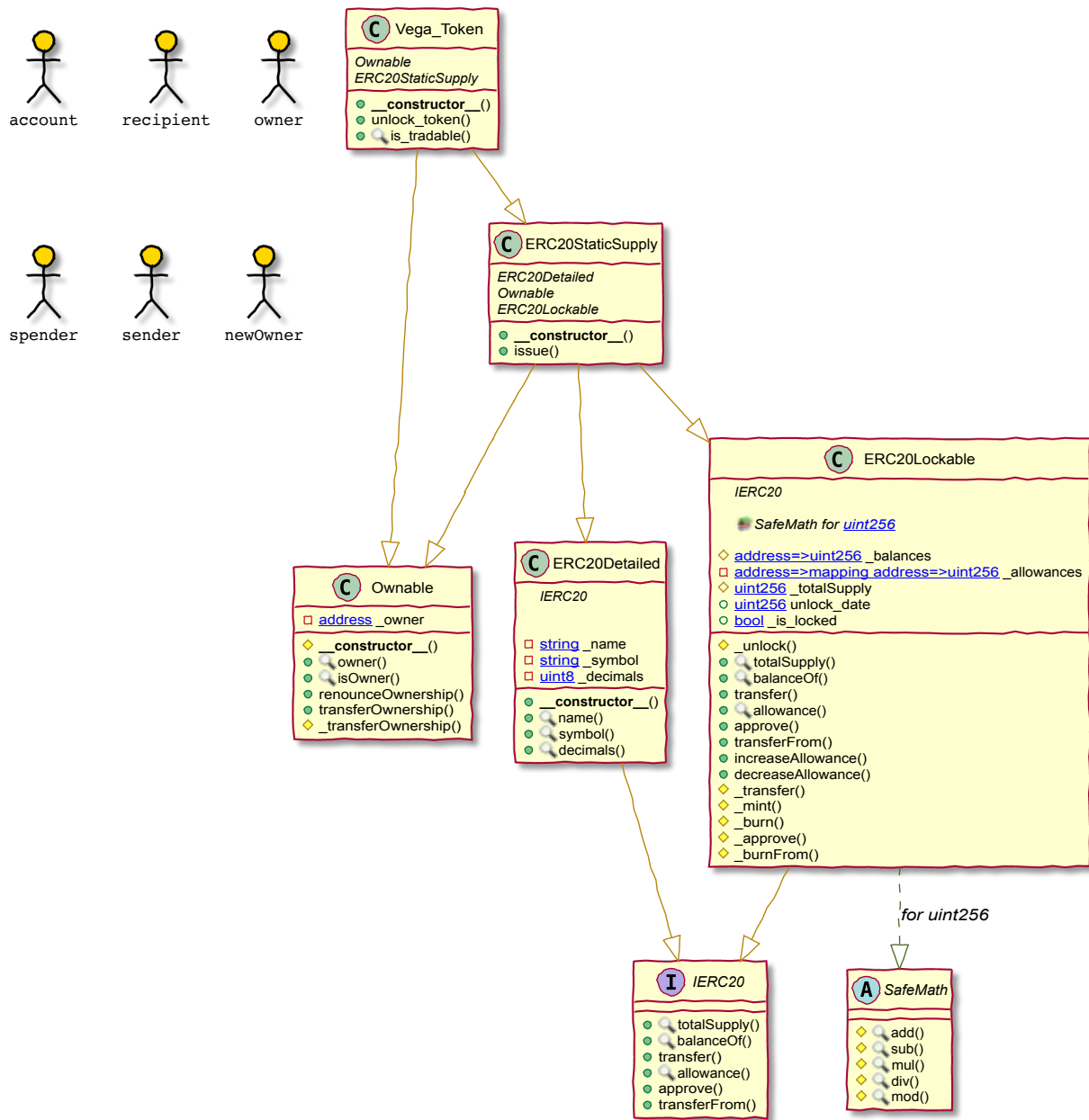
Inheritance graph

Call Graph



Function call graph and contract interaction

Vega Token



Vega Token Outline

4 Key Observations/Recommendations

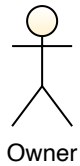
- The following libraries are part of the project workspace but not referenced by the Vega Token:
 - o [redacted]
 - o [redacted] and therefore also [redacted] which is only referenced by [redacted]
 - o **Acknowledged and fixed by the development team.**
- [SWC-103](#): The codebase sets compiler flags ([redacted]) that allows compilation with any solidity 0.5.x compiler potentially allowing to fall back to

untested versions or compiler versions with known bugs, vulnerabilities or fewer security checks. It is recommended to lock the version to the solidity compiler version the codebase has been thoroughly tested.

- Please note that the latest solidity compiler version released is [v0.6.2](#).
- **Acknowledged and fixed by the development team.**
- [REDACTED] should be declared [REDACTED] and all uppercase [REDACTED].
 - **Acknowledged and fixed by the development team.** (v. [Issue 6.1](#))
- [REDACTED] could use [REDACTED] instead of duplicating code.
 - **Acknowledged and fixed by the development team.**
- The contract [REDACTED] should be named using CapWords as recommended by the [solidity style guide](#).
 - **Acknowledged and fixed by the development team.**

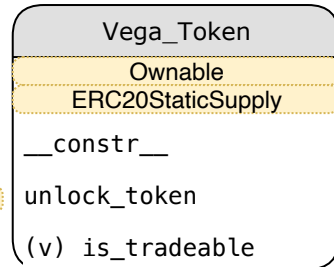
5 Security Specification

This section describes the behavior of the system under audit from a security perspective. It is best combined with the overview given in [section 3 - System Overview](#). Please note that this document is not a substitute for documentation. The purpose of this section is to identify specific security properties that were validated by the audit team. Furthermore, the information contained in this section can be used for internal security activities and we recommend documenting and building-upon the trust model that has been established.



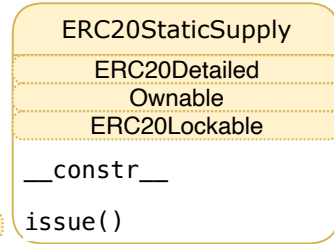
- initially assigned to contract deployer
 - can transfer ownership
 - can renounce ownership
- Owner
- is initially in control of all tokens
 - issues tokens to users at will (completely centralized)
 - can issue while locked

owner



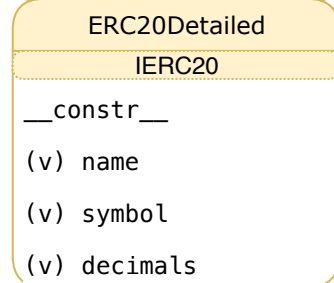
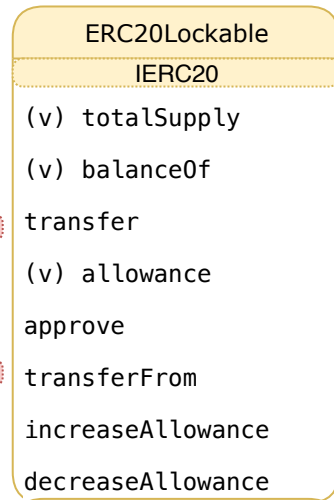
- can transfer tokens
 - if contract is unlocked
 - or time past unlock_time (auto unlock)
 - can set allowance even before that
- user

owner



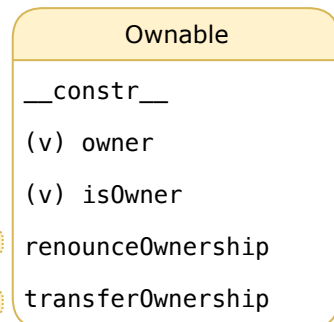
onlyUnlocked

onlyUnlocked



owner

owner



Permission and Actors overview

5.1 Actors

The relevant actors are as follows:

- **Deployer** - initially deploys the system.

- **Owner** - privileged account that is used to distribute tokens.
- **Token Holder** - address with a token balance > 0.
- **Any Ethereum Account** - address with no token balance.

5.2 Trust Model

In any smart contract system, it's important to identify what trust is expected/required between various actors. For this audit, we established the following trust model:

- **Deployer** initially deploys the smart contract system. [REDACTED] is [REDACTED] and therefore turns the **Deployer** into the **Owner** in control of the Token.
 - An amount of tokens is initially minted and assigned to the [REDACTED] contract address.
 - New tokens cannot be minted after the deployment phase.
 - Existing tokens cannot be burned.
 - Token Supply is static throughout the lifetime of the token.
 - Transfers for **Token Holders** of the [REDACTED] are initially locked.
 - Token transfers are automatically unlocked after a hardcoded [REDACTED].
- **Owner** can manually unlock the token before [REDACTED] has passed, allowing **Token Holders** to start transferring their tokens.
- **Owner** can issue any amount of tokens owned by the [REDACTED] contract address to **Token Holder** or **Any Ethereum Account**.
- **Owner** can issue tokens after the [REDACTED] has passed.
- **Owner** can renounce ownership of the contract leaving the contract unmanaged.
- **Owner** can transfer ownership of the contract to **Any Ethereum Account**.
- **Owner** cannot burn tokens.
- **Owner** cannot mint new tokens and therefore change supply.
- **Owner** can unlock token transfers immediately after deployment.
- **Token Holders** can transfer tokens to **Any Ethereum Account** if the token is unlocked (either [REDACTED] has passed or token is manually unlocked by **Owner**).
- **Token Holders** can approve and change token approvals also before the token trading is unlocked.

5.3 Important Security Properties

The following is a non-exhaustive list of security properties that were verified in this audit:

- `Token`, `Token`, `Token` and `Token` is constant throughout the lifetime of the Token.
 - The sum of all tokens in circulations equals the `Token` set for the token.
 - Only the **Owner** can mint tokens.
 - Tokens are only minted and assigned to the `Token` contract address when creating the contract.
- **Token Holders** are in control of their token balances. An administrative account (**Owner**) cannot interfere (transfer/burn) with **Token Holder**'s token balances.
- Only the **Owner** can transfer/renounce ownership of the contract.
 - Transferring ownership allows the new owner to issue tokens on behalf of the `Token` contract or unlock the token if it is not yet unlocked.
- Renouncing ownership removes **Owner** ability to manage the contract.
 - Former **Owner** loses the ability to assign tokens held by the `Token` contract to new **Token Holders**.
 - Former **Owner** loses the ability to manually unlock the token.
 - A new **Owner** cannot be assigned after renouncing ownership.
 - If the ownership is renounced while the contract is still locked, **Token Holders** can exercise their right to transfer tokens **but only after** `Token` has passed which automatically unlocks the token.
- The contract cannot be selfdestruct.

6 Issues

Each issue has an assigned severity:

- **Minor** issues are subjective in nature. They are typically suggestions around best practices or readability. Code maintainers should use their own judgment as to whether to address such issues.
- **Medium** issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
- **Major** issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be

addressed.

- **Critical** issues are directly exploitable security vulnerabilities that need to be fixed.

6.1 ERC20Lockable - inconsistent locking status Minor ✓ Fixed

Resolution

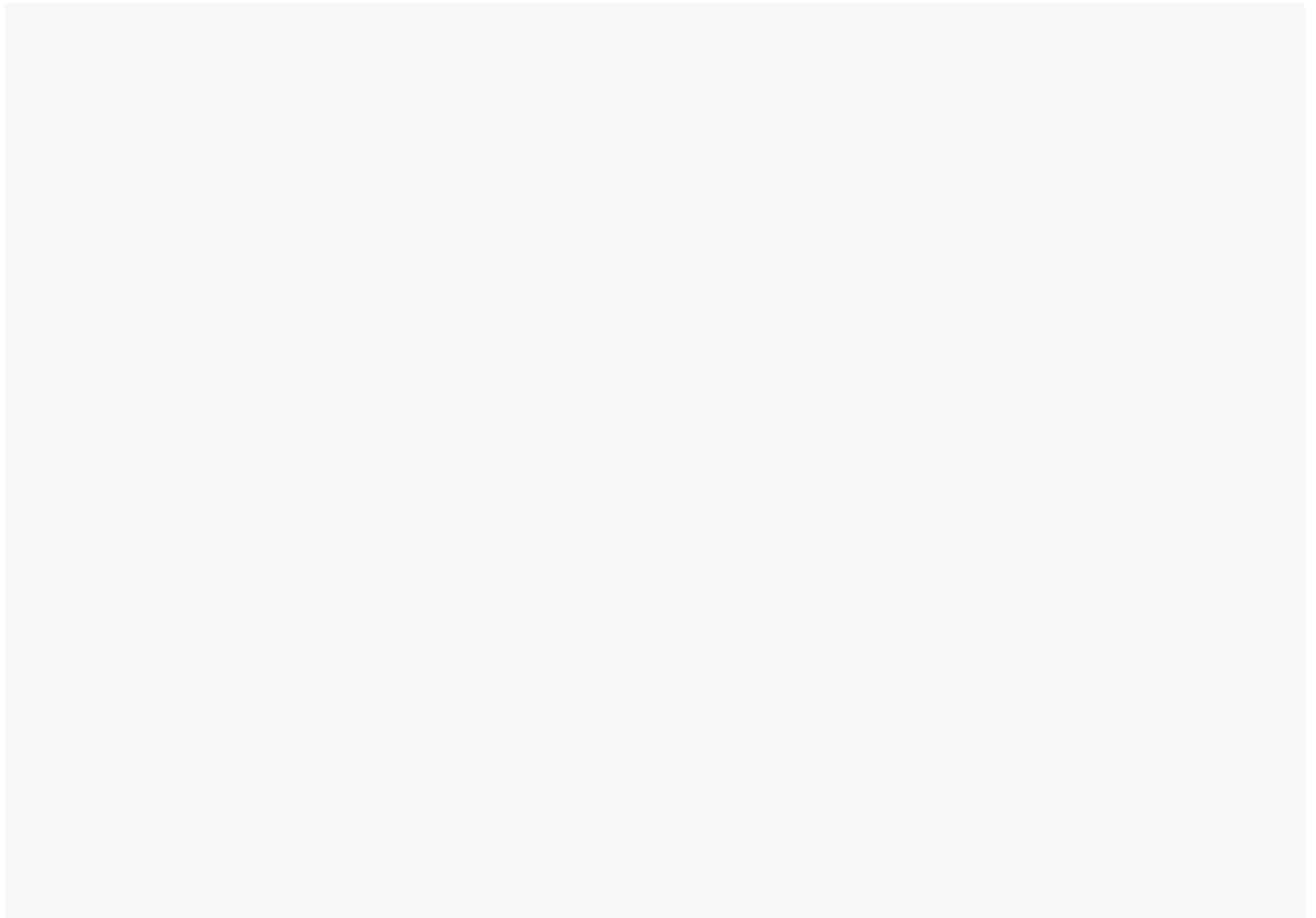
Issue was fixed by completely removing the unlock date mechanism.

Description

_____ will incorrectly return _____ if the token is never manually unlocked by the owner but _____ has passed, which will automatically unlock trading.

Examples

code/ERC20Lockable.sol:L48-L67



Recommendation

- declare `lock` as `bool` instead of `bool`
- create a getter method that correctly returns the locking status

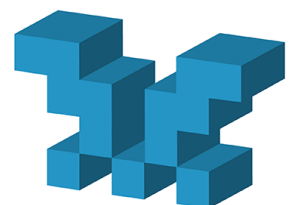
- make `lock` use the newly created getter
(`lock`)
- make `lock` use the newly created getter
(`lock`)
- `lock` should raise an errorcondition when called on an already unlocked contract
 - it could make sense to emit a “contract has been unlocked” event for auditing purposes

7 Tool-Based Analysis

Several tools were used to perform automated analysis of the reviewed contracts. These issues were reviewed by the audit team, and relevant issues are listed in the Issue Details section.

7.1 MythX

MythX is a security analysis API for Ethereum smart contracts. It performs multiple types of analysis, including fuzzing and symbolic execution, to detect many common vulnerability types. The tool was used for automated vulnerability discovery for all audited contracts and libraries. More details on MythX can be found at mythx.io.



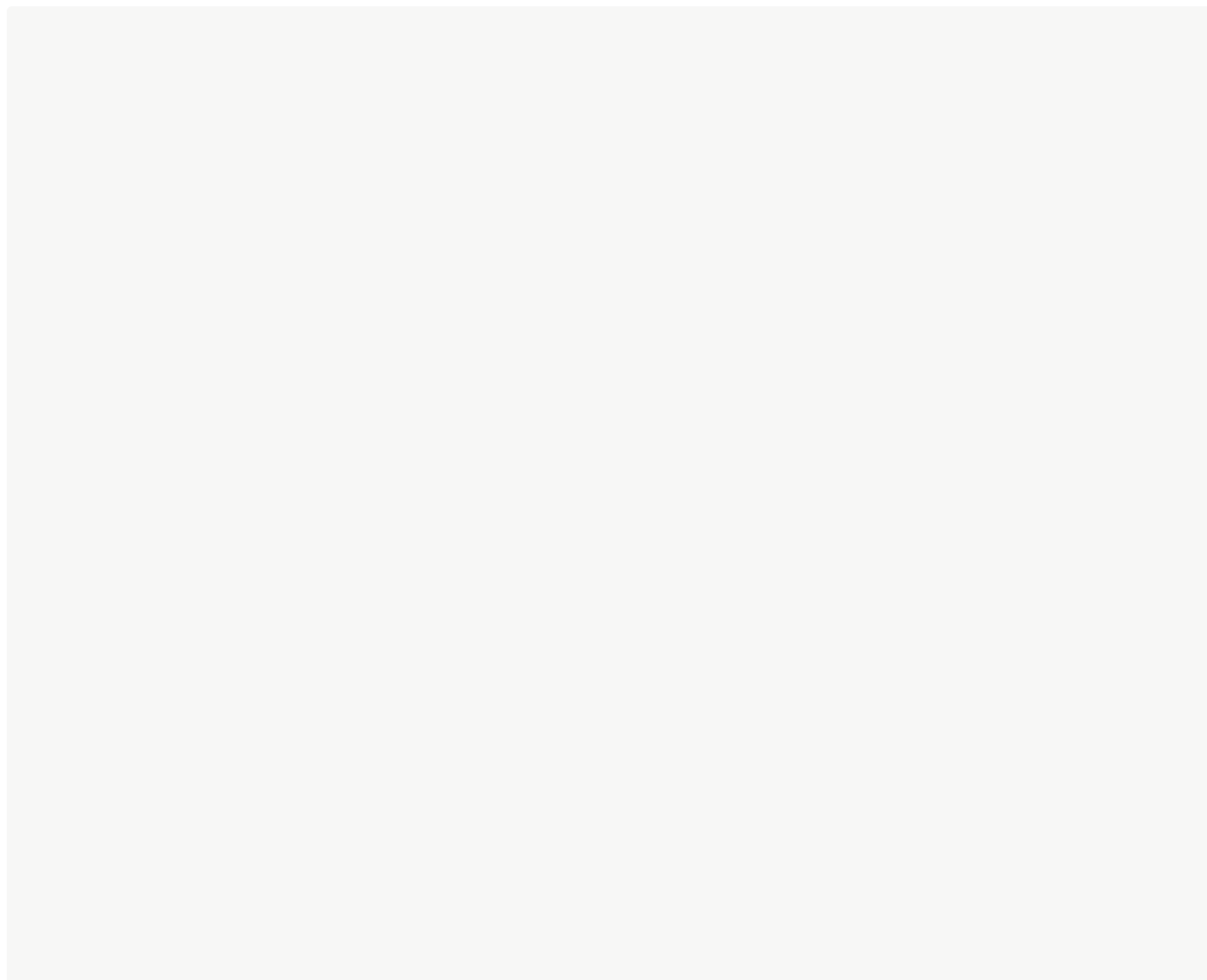
The output of a MythX Full Mode analysis was reviewed by the audit team and no relevant issues were raised as part of the process.

7.2 Ethlint

[Ethlint](#) is an open source project for linting Solidity code. Only security-related issues were reviewed by the audit team.



Below is the raw output of the Ethlint vulnerability scan:



7.3 Surya

Surya is a utility tool for smart contract systems. It provides a number of visual outputs and information about the structure of smart contracts. It also supports querying the function call graph in multiple ways to aid in the manual inspection and control flow analysis of contracts.

Below is a complete list of functions with their visibility and modifiers:
















Sūrya's Description Report






















Files Description Table













File Name	SHA-1 Hash
contracts/Vega_Token.sol	b92b3c54b2f47a88fa9e84534046b462dbae9aa
contracts/Address.sol	1213b0f150dd5e3f694c3721c44cb5cc3202b743
contracts/ERC20Detailed.sol	7e4d00c462120565201f28361b29201d1bfe0a34
contracts/IERC20.sol	72c15b6a16b7dc92e69ff97ccfe1958d9948e200
contracts/ERC20Lockable.sol	377447995444beee2b3c5342bfa9b1bbc1d08356
contracts/SafeMath.sol	c8bda5eb19c16d34bc48bf115229a9b967feb6ef
contracts/ERC20StaticSupply.sol	bf3e66af74470eed08d0e0f82b9a98705a745c7c
contracts/Ownable.sol	12ec51ec8a3b4eed6326434fd0f5926b40602778
contracts/Roles.sol	2c85acf184ae36f96ebafd8f6e26232ea459a711
contracts/SafeERC20.sol	ebd65ea9a0cdbc29bbbf651a1076d51be031443

Contracts Description Table



Contract	Type	Bases	Visibility	Mutability	Modifiers
L	Function Name				
Vega_Token	Implementation	Ownable, ERC20StaticSupply			
L		Public !		ERC20Stat	
L	unlock_token	Public !		onlyOv	
L	is_tradable	Public !		NO	
Address	Library				
L	isContract	Internal			
L	toPayable	Internal			
ERC20Detailed	Implementation	IERC20			

Contract	Type	Bases		
L		Public !		NO
L	name	Public !		NO
L	symbol	Public !		NO
L	decimals	Public !		NO
IERC20	Interface			
L	totalSupply	External !		NO
L	balanceOf	External !		NO
L	transfer	External !		NO
L	allowance	External !		NO
L	approve	External !		NO
L	transferFrom	External !		NO
ERC20Lockable	Implementation	IERC20		
L	_unlock	Internal 		
L	totalSupply	Public !		NO
L	balanceOf	Public !		NO
L	transfer	Public !		onlyUnl
L	allowance	Public !		NO
L	approve	Public !		NO
L	transferFrom	Public !		onlyUnl
L	increaseAllowance	Public !		NO
L	decreaseAllowance	Public !		NO
L	_transfer	Internal 		
L	_mint	Internal 		
L	_burn	Internal 		
L	_approve	Internal 		
L	_burnFrom	Internal 		

Contract	Type	Bases		
SafeMath	Library			
L	add	Internal 		
L	sub	Internal 		
L	sub	Internal 		
L	mul	Internal 		
L	div	Internal 		
L	div	Internal 		
L	mod	Internal 		
L	mod	Internal 		
ERC20StaticSupply	Implementation	ERC20Detailed, Ownable, ERC20Lockable		
L		Public !		ERC20D
L	issue	Public !		onlyOv
Ownable	Implementation			
L		Internal 		
L	owner	Public !		NO
L	isOwner	Public !		NO
L	renounceOwnership	Public !		onlyOv
L	transferOwnership	Public !		onlyOv
L	_transferOwnership	Internal 		
Roles	Library			
L	add	Internal 		
L	remove	Internal 		
L	has	Internal 		
SafeERC20	Library			

Contract	Type	Bases		
L	safeTransfer	Internal 		
L	safeTransferFrom	Internal 		
L	safeApprove	Internal 		
L	safeIncreaseAllowance	Internal 		
L	safeDecreaseAllowance	Internal 		
L	callOptionalReturn	Private 		

Legend

Symbol	Meaning
	Function can modify state
	Function is payable

Appendix 1 - Disclosure

ConsenSys Diligence ("CD") typically receives compensation from one or more clients (the "Clients") for performing the analysis contained in these reports (the "Reports"). The Reports may be distributed through other means, including via ConsenSys publications and other distributions.

The Reports are not an endorsement or indictment of any particular project or team, and the Reports do not guarantee the security of any particular project. This Report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any Third-Party in any respect, including regarding the bugfree nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. No third party should rely on the Reports in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. Specifically, for the avoidance of doubt, this Report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. CD owes no duty to any Third-Party by virtue of publishing these Reports.

PURPOSE OF REPORTS The Reports and the analysis described therein are created solely for Clients and published with their consent. The scope of our review is limited to a review of Solidity code and only the Solidity code we note as being within the scope of our review within this report. The Solidity language itself remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond Solidity that could present security risks. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty.

CD makes the Reports available to parties other than the Clients (i.e., "third parties") – on its website. CD hopes that by making these analyses publicly available, it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.

LINKS TO OTHER WEB SITES FROM THIS WEB SITE You may, through hypertext or other computer links, gain access to web sites operated by persons other than ConsenSys and CD. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that ConsenSys and CD are not responsible for the content or operation of such Web sites, and that ConsenSys and CD shall have no liability to you or any other person or entity for the use of third party Web sites. Except as described below, a hyperlink from this web Site to another web site does not imply or mean that ConsenSys and CD endorses the content on that Web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the Reports. ConsenSys and CD assumes no responsibility for the use of third party software on the Web Site and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

TIMELINESS OF CONTENT The content contained in the Reports is current as of the date appearing on the Report and is subject to change without notice. Unless indicated otherwise, by ConsenSys and CD.