CERTIK

Code Security Assessment

Shentu Security Oracle

Oct 11th, 2020

Table of Contents

Summary

This report has been prepared for Shentu Security Oracle to discover issues and vulnerabilities in the source code of the Shentu Security Oracle 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	Shentu Security Oracle
Platform	Ethereum
Language	Solidity
Codebase	
Commit	

Audit Summary

Delivery Date	Oct 11, 2020
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Mitigated	Resolved
● Critical	0	0	0	0	0	0	0
● Major	0	0	0	0	0	0	0
● Medium	0	0	0	0	0	0	0
● Minor	0	0	0	0	0	0	0
● Informational	14	0	0	2	0	0	12
● Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum

Executive Summary

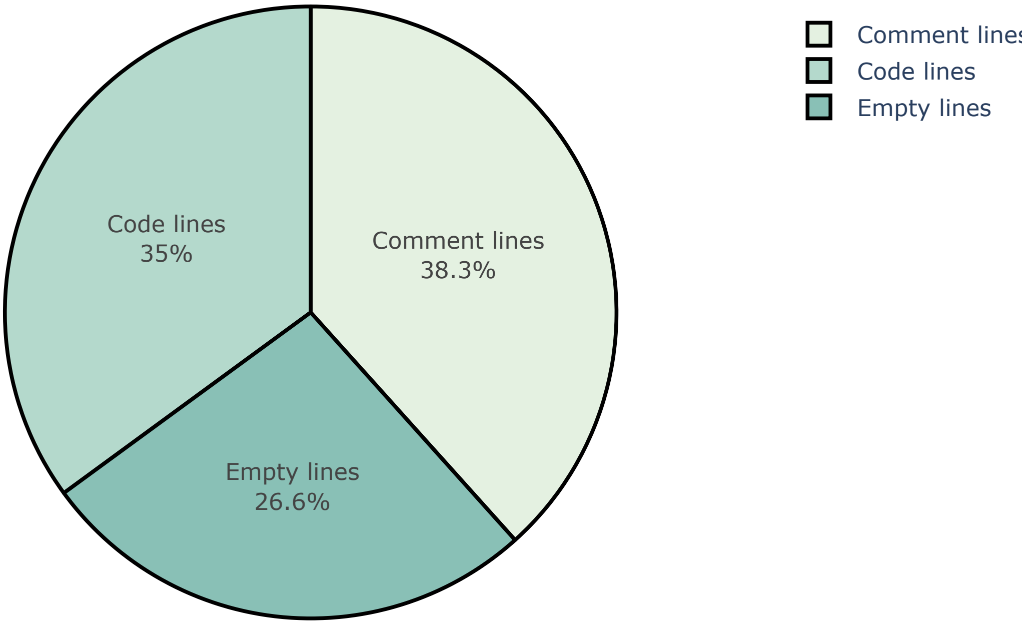
The report represents the results of our engagement with the Shentu Chain on their Security Oracle

The high severity exhibits stem from the different compiler versions that the codebase allowed and one referenced vulnerability, which were immediately refactored. In contrast, the lower severity ones mainly refer to optimization and Solidity coding standards.

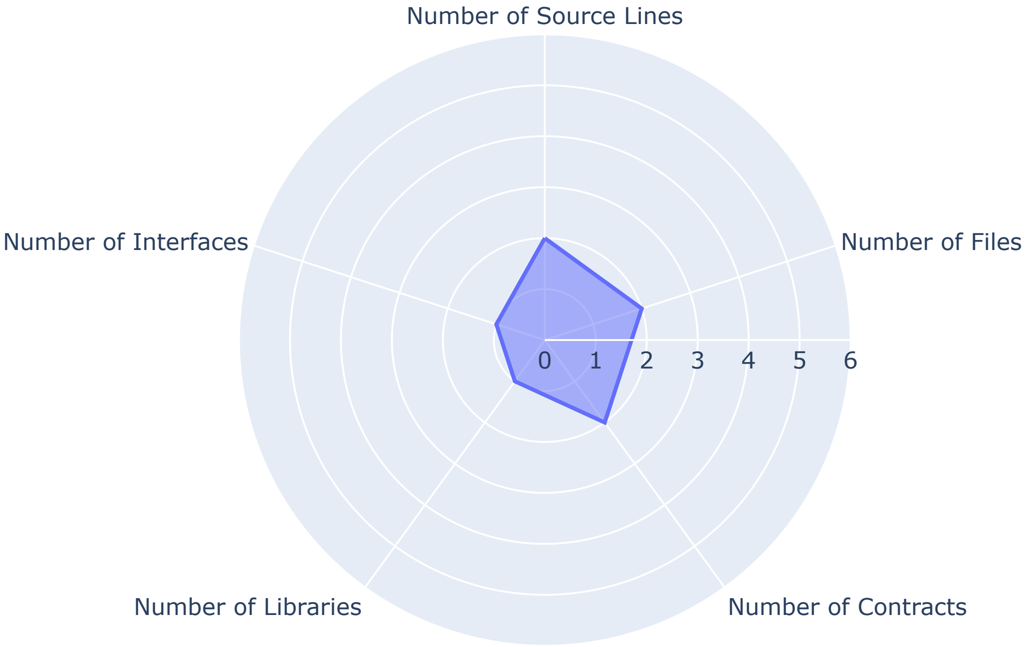
Hence the codebase can be deemed to be of high security and quality.

Diagrams

Source Line Chart



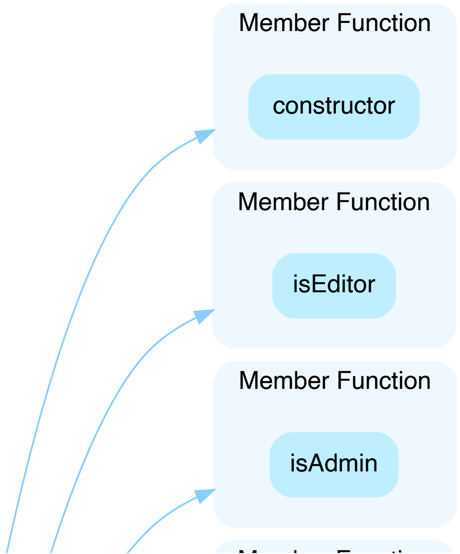
Summary Chart

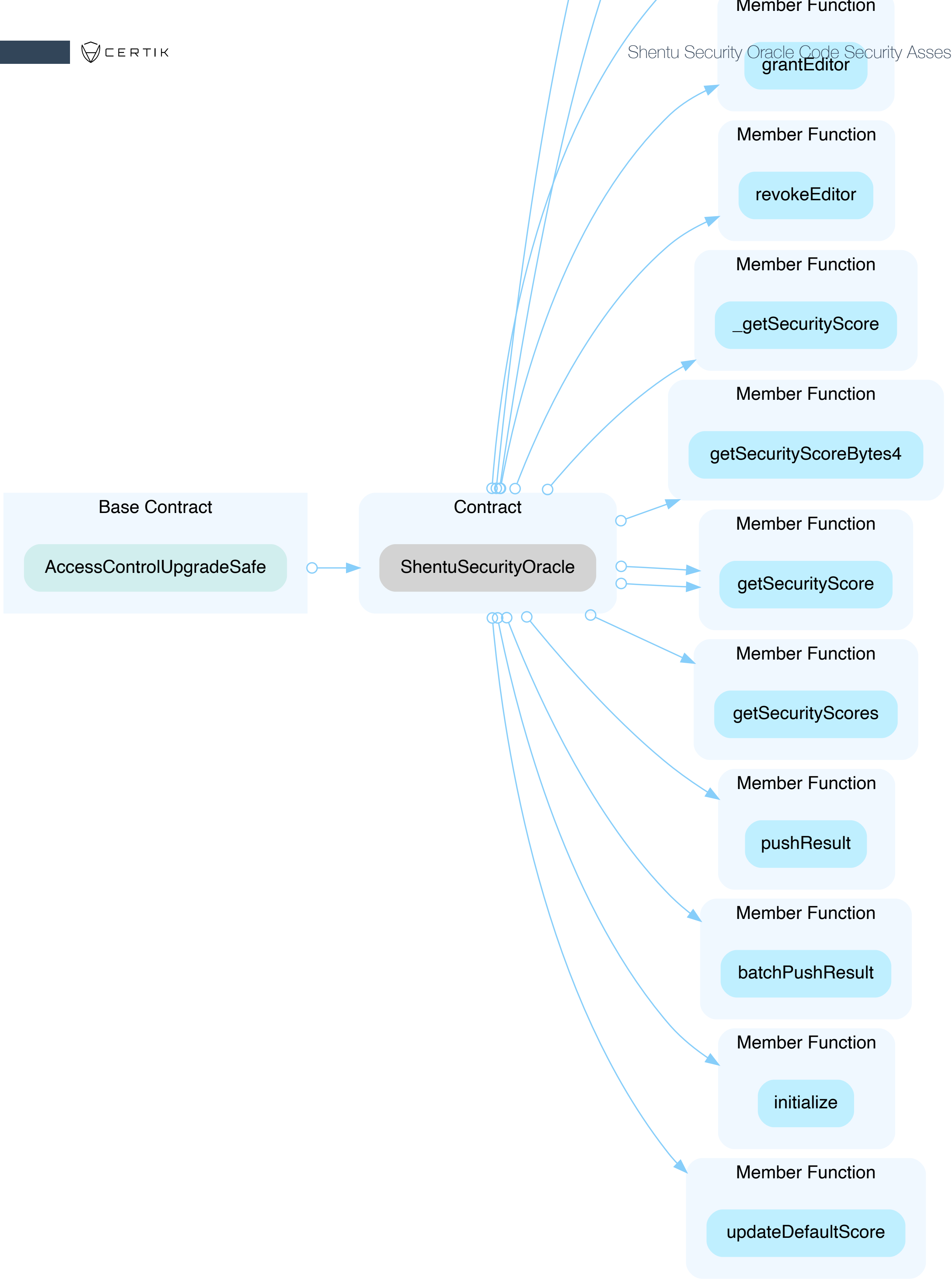


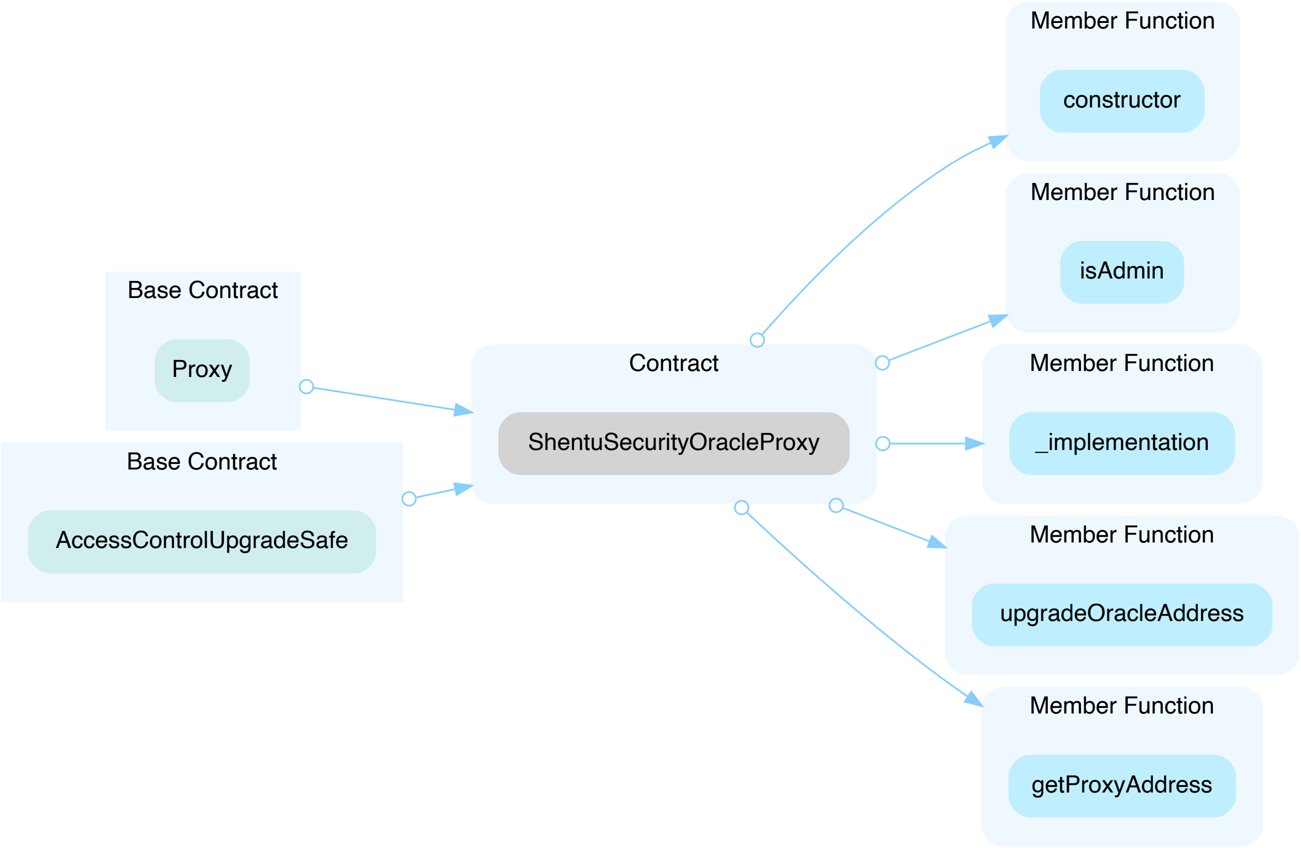
Function	*# of Invocations*	*Invocation Location*
isEditor	5	ShentuSecurityOracle.sol: 50, ShentuSecurityOracle.sol: 54, openzeppelin/AccessControl.sol: 133, openzeppelin/AccessControl.sol: 148, ShentuSecurityOracleProxy.sol: 29
isAdmin	2	ShentuSecurityOracle.sol: 38, ShentuSecurityOracle.sol: 45
_getSecurityScore	2	ShentuSecurityOracle.sol: 89, ShentuSecurityOracle.sol: 109
getSecurityScoreBytes4	2	ShentuSecurityOracle.sol: 97, ShentuSecurityOracle.sol: 126
pushResult	2	openzeppelin/EnumerableSet.sol: 55, openzeppelin/EnumerableSet.sol: 220
initialize	2	openzeppelin/Proxy.sol: 64, openzeppelin/Proxy.sol: 72
_add	2	ShentuSecurityOracle.sol: 30, ShentuSecurityOracleProxy.sol: 11
_remove	2	openzeppelin/AccessControl.sol: 135, openzeppelin/AccessControl.sol: 190
_contains	2	openzeppelin/AccessControl.sol: 150, openzeppelin/AccessControl.sol: 170
_length	1	ShentuSecurityOracle.sol: 38
_at	1	ShentuSecurityOracle.sol: 164
isConstructor	1	ShentuSecurityOracle.sol: 33
_delegate	1	openzeppelin/EnumerableSet.sol: 203
_implementation	1	openzeppelin/EnumerableSet.sol: 213
_fallback	1	openzeppelin/EnumerableSet.sol: 227
_beforeFallback	1	openzeppelin/EnumerableSet.sol: 241
__AccessControl_init_unchained	1	openzeppelin/Initializable.sol: 34
hasRole	1	openzeppelin/Proxy.sol: 56
grantRole	1	openzeppelin/Proxy.sol: 56
revokeRole	1	openzeppelin/Proxy.sol: 55
_setupRole	1	openzeppelin/AccessControl.sol: 44
_setRoleAdmin	1	ShentuSecurityOracle.sol: 58
_grantRole	1	ShentuSecurityOracle.sol: 62
_revokeRole	1	ShentuSecurityOracle.sol: 31
__Context_init_unchained	1	openzeppelin/Context.sol: 21
isAdmin	1	ShentuSecurityOracleProxy.sol: 22

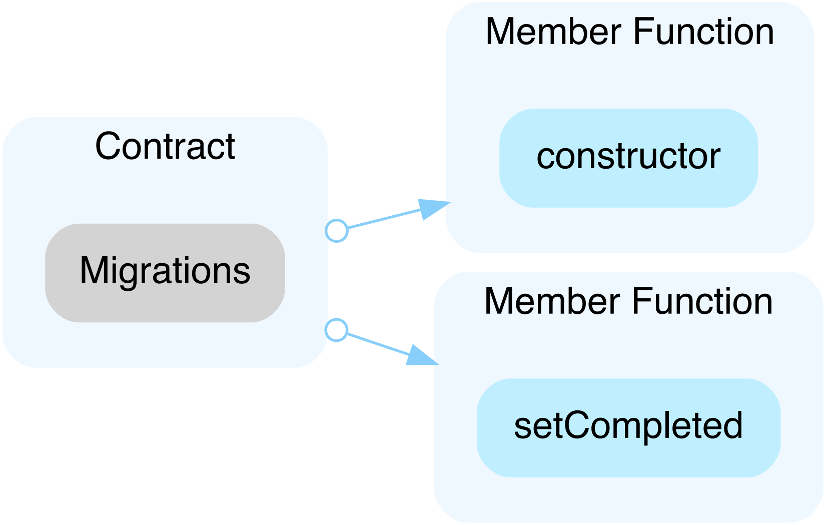
Contract	*# of Invocations*	*Invocation Location*
ShentuSecurityOracle	0	
Migrations	0	
EnumerableSet	0	
Initializable	0	
Proxy	0	
AccessControlUpgradeSafe	0	
ContextUpgradeSafe	0	
Address	0	
ShentuSecurityOracleProxy	0	
SecurityOracle	0	
DeFiExample	0	

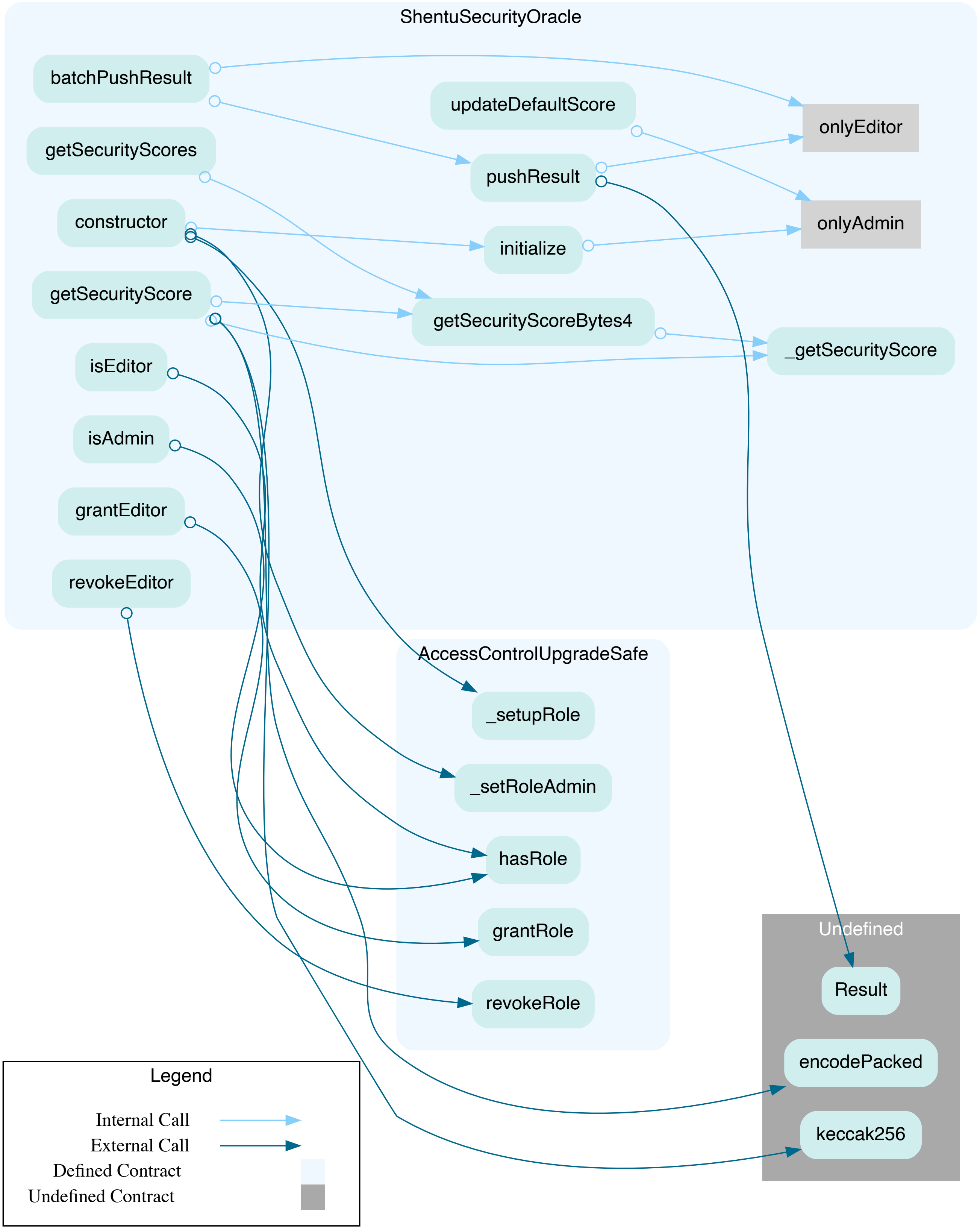
Function	*# of Invocations*	*Invocation Location*
constructor	0	
grantEditor	0	
revokeEditor	0	
getSecurityScore	0	
getSecurityScore	0	
getSecurityScores	0	
batchPushResult	0	
updateDefaultScore	0	
constructor	0	
setCompleted	0	
add	0	
remove	0	
contains	0	
length	0	
at	0	
fallback	0	
receive	0	
__AccessControl_init	0	
getRoleMemberCount	0	
getRoleMember	0	
getRoleAdmin	0	
renounceRole	0	
__Context_init	0	
_msgSender	0	
_msgData	0	
isContract	0	
sendValue	0	
constructor	0	
_implementation	0	
upgradeOracleAddress	0	
getProxyAddress	0	
getSecurityScore	0	
getSecurityScore	0	
getSecurityScoreBytes4	0	
getSecurityScores	0	
constructor	0	
callGetSecurityScore	0	
callGetSecurityScoreBytes4	0	
callGetSecurityScores	0	

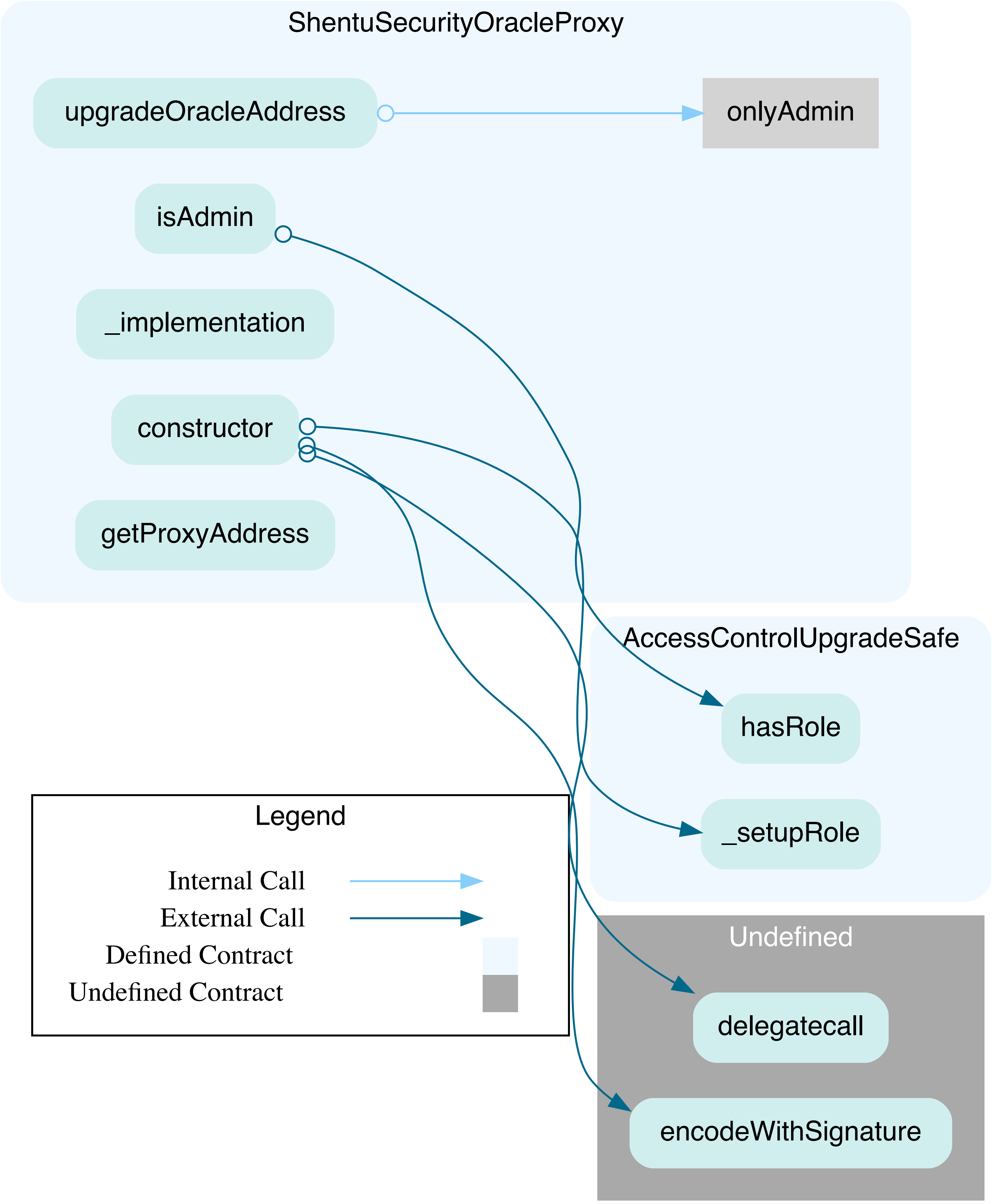


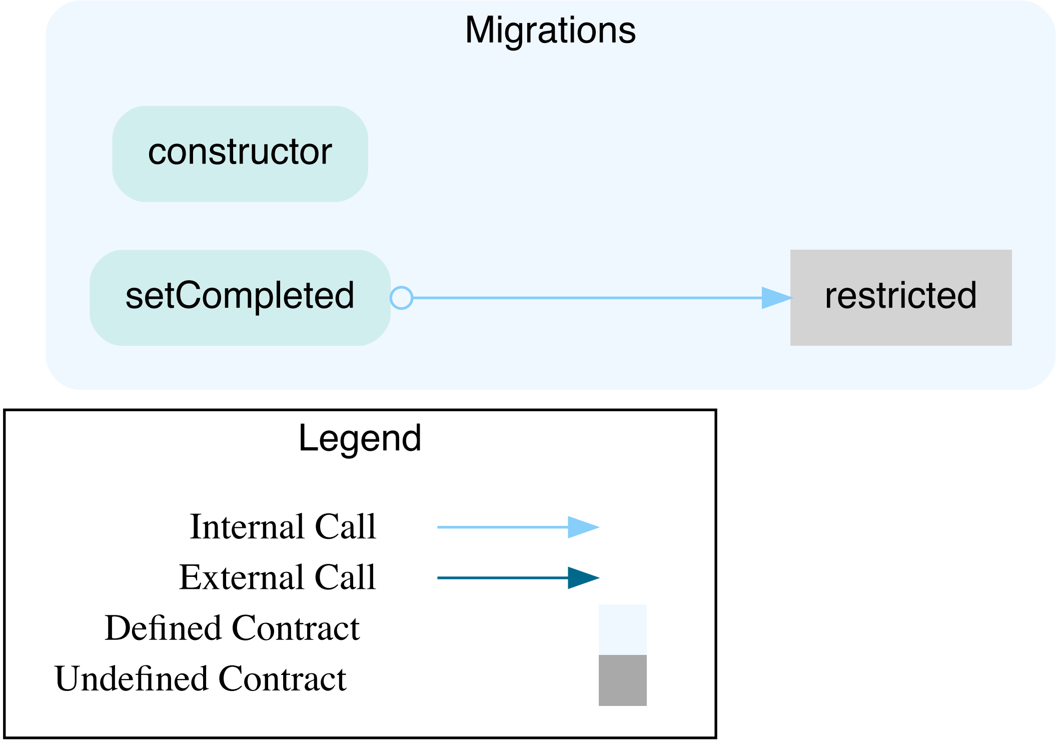

Member Function
grantEditor
Member Function
revokeEditor
Member Function
_getSecurityScore
Member Function
getSecurityScoreBytes4
Base Contract
AccessControlUpgradeSafe
Contract
ShentuSecurityOracle
Member Function
getSecurityScore
Member Function
getSecurityScores
Member Function
pushResult
Member Function
batchPushResult
Member Function
initialize
Member Function
updateDefaultScore

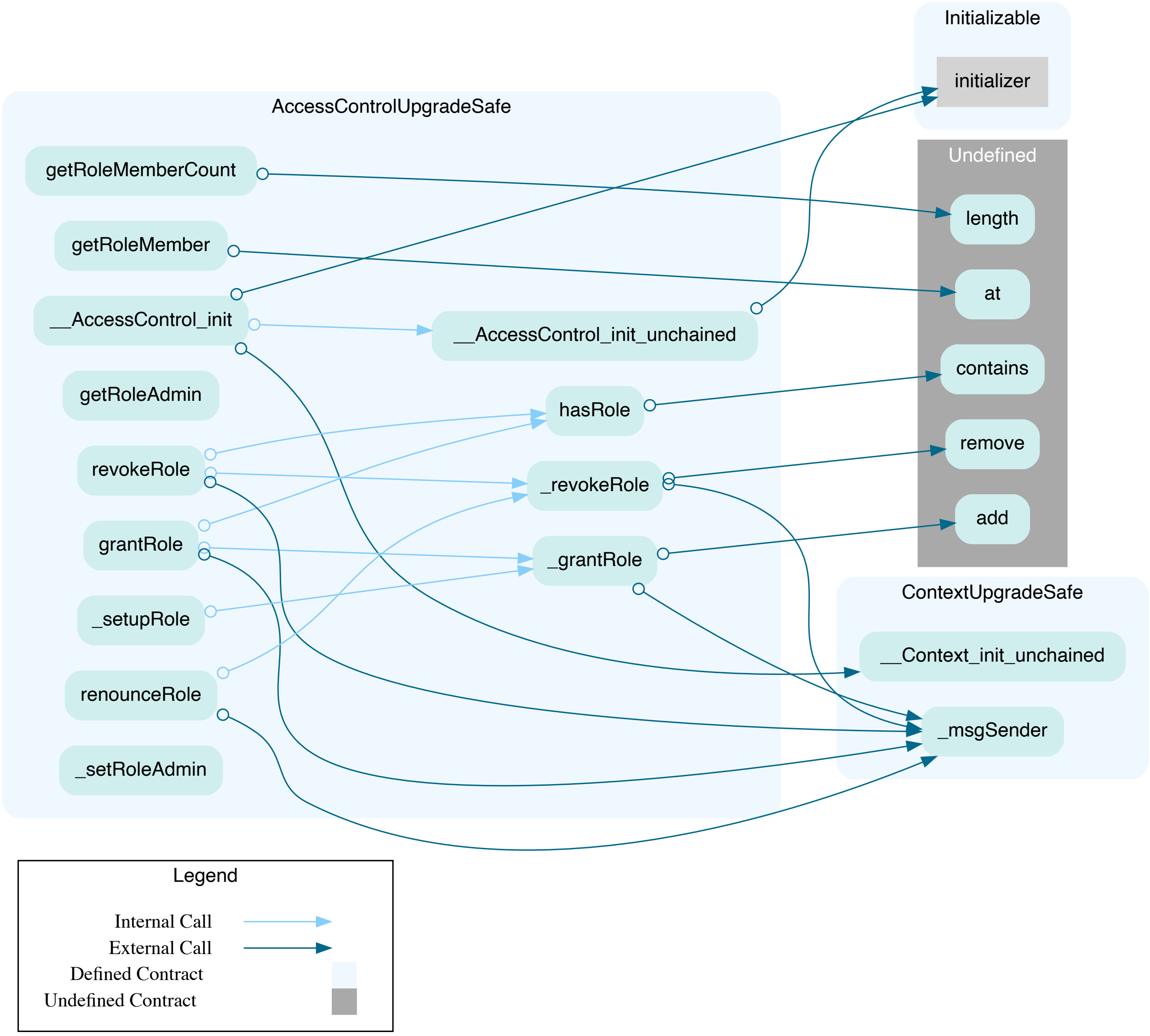

Base Contract
Proxy
Base Contract
AccessControlUpgradeSafe
Contract
ShentuSecurityOracleProxy
Member Function
constructor
Member Function
isAdmin
Member Function
_implementation
Member Function
upgradeOracleAddress
Member Function
getProxyAddress


Contract
Migrations
Member Function
constructor
Member Function
setCompleted


ShentuSecurityOracle
batchPushResult
updateDefaultScore
onlyEditor
getSecurityScores
pushResult
onlyAdmin
constructor
initialize
getSecurityScore
getSecurityScoreBytes4
_getSecurityScore
isEditor
isAdmin
grantEditor
revokeEditor
AccessControlUpgradeSafe
_setupRole
_setRoleAdmin
hasRole
grantRole
revokeRole
Undefined
Result
encodePacked
keccak256
Legend
Internal Call
External Call
Defined Contract
Undefined Contract


ShentuSecurityOracleProxy
upgradeOracleAddress
onlyAdmin
isAdmin
_implementation
constructor
getProxyAddress
AccessControlUpgradeSafe
hasRole
_setupRole
Undefined
delegatecall
encodeWithSignature
Legend
Internal Call
External Call
Defined Contract
Undefined Contract


Migrations
constructor
setCompleted
restricted
Legend
Internal Call
External Call
Defined Contract
Undefined Contract



Function api table for contract ShentuSecurityOracle

Function Name	***Visibility***	***State Variable Modification***	***Modifier***
isEditor_1	public	-	-
isAdmin_1	public	-	-
grantEditor_1	public	-	-
revokeEditor_1	public	-	-
getSecurityScoreBytes4_2	public	-	-
getSecurityScore_2	public	-	-
getSecurityScore_1	public	-	-
getSecurityScores_2	public	-	-
pushResult_4	public	_results	onlyEditor
batchPushResult_4	public	-	onlyEditor
initialize_0	public	defaultScore	onlyAdmin
updateDefaultScore_1	public	defaultScore	onlyAdmin

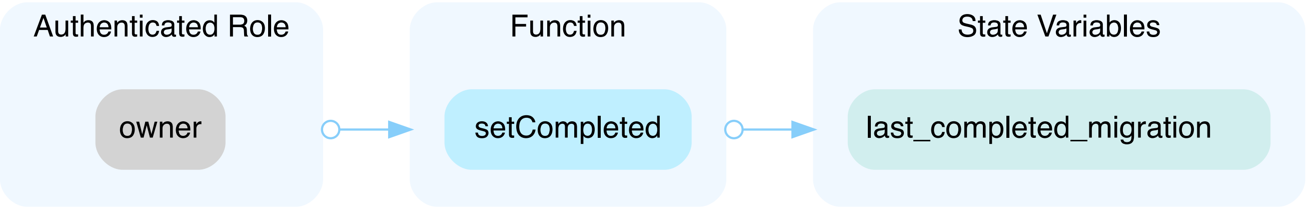
Function api table for contract ShentuSecurityOracleProxy

Function Name	***Visibility***	***State Variable Modification***	***Modifier***
isAdmin_1	public	-	-
upgradeOracleAddress_1	public	currentOracleAddress	onlyAdmin
getProxyAddress_0	public	-	-

Function api table for contract SecurityOracle

Function Name	***Visibility***	***State Variable Modification***	***Modifier***
getSecurityScore_1	external	-	-
getSecurityScore_2	external	-	-
getSecurityScoreBytes4_2	external	-	-
getSecurityScores_2	external	-	-

State Variable	***Variable Visibility***	***Functions Modifying the Variable***
_results	private	pushResult_4 (public, ShentuSecurityOracle.sol)
defaultScore	public	initialize_0 (public, ShentuSecurityOracle.sol)
defaultScore	public	updateDefaultScore_1 (public, ShentuSecurityOracle.sol)
owner	public	constructor (public, Migrations.sol)
last_completed_migration	public	setCompleted_1 (public, Migrations.sol)
currentOracleAddress	public	constructor (public, ShentuSecurityOracleProxy.sol)
currentOracleAddress	public	upgradeOracleAddress_1 (public, ShentuSecurityOracleProxy.sol)
_securityOracleAddress	private	constructor (public, DeFiExample.sol)



Findings

14
Total Issues

🟥	**Critical**	**0** (0.00%)
🟧	**Major**	**0** (0.00%)
🟨	**Medium**	**0** (0.00%)
🟨	**Minor**	**0** (0.00%)
🟦	**Informational**	**14** (100.00%)
🟩	**Discussion**	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Unlocked Compiler Version	Language Specific	● Informational	✓ Resolved
DFE-01	Unlocked Compiler Versions	Language Specific	● Informational	✓ Resolved
DFE-02	Different Compiler Versions	Language Specific	● Informational	✓ Resolved
SSO-01	Unlocked Compiler Version	Language Specific	● Informational	✓ Resolved
SSO-02	`struct` Optimization	Optimization	● Informational	✓ Resolved
SSO-03	Uncommon Naming Convention	Coding Style	● Informational	✓ Resolved
SSO-04	Use of `memory` Over `storage`	Optimization	● Informational	✓ Resolved
SSO-05	Potential High `gas` Operation	Optimization	● Informational	ⓘ Acknowledged
SSO-06	Check Against the Zero Address	Volatile Code	● Informational	✓ Resolved
SSO-07	Different Compiler Versions	Language Specific	● Informational	✓ Resolved
SSO-08	Malicious Hash Collision	Language Specific	● Informational	✓ Resolved
SSO-09	`initialize` Paradigm	Language Specific	● Informational	ⓘ Acknowledged
SSP-01	Uncommon Naming Convention	Coding Style	● Informational	✓ Resolved
SSP-02	Different Compiler Versions	Language Specific	● Informational	✓ Resolved

GLOBAL-01 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	● Informational	Global	✓ Resolved

Description

If the compiler version is between `0.4.21` and `0.4.26`, then the contract raises a compilation error due to the keyword `payable`.

Recommendation

We advise that compiler versions below `0.5.0` should be avoided

Alleviation

The team opted to consider our references and changed to compiler version `0.5.17`.

DFE-01 | Unlocked Compiler Versions

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/DeFiExample.sol (a09e939): 2	✓ Resolved

Description

An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers. This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the full project can be compiled at.

Alleviation

The team opted to consider our references and changed to compiler version 0.5.17 .

DFE-02 | Different Compiler Versions

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/DeFiExample.sol (a09e939): 2	✓ Resolved

Description

If the compiler version is between `0.4.21` and `0.4.26`, then the contract raises a compilation error.

Recommendation

We advise that compiler versions below `0.5.0` should be avoided.

Alleviation

The team opted to consider our references and changed to compiler version `0.5.17`.

SSO-01 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939)	✓ Resolved

Description

An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers.This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the full project can be compiled at.

Alleviation

The team opted to consider our references and changed to compiler version `0.5.17`.

SSO-02 | `struct` Optimization

Category	Severity	Location	Status
Optimization	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 17 ~20	✓ Resolved

Description

Each struct packs its members in 256-bit chunks. The Result struct contains the score (uint8) and the expiration (uint256) members, thus reserving two 256-bit chunks in storage.

Recommendation

We advise that the data type of the expiration member of the Result struct is changed to `uint248`, as the maximum bit-size that a datetime variable reserves are 64-bit, resulting in a one chuck storage reservation for the struct.

Alleviation

The team opted to consider our references and changed the data type of the expiration member of the Result struct to `uint248`.

SSO-03 | Uncommon Naming Convention

Category	Severity	Location	Status
Coding Style	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 25	✓ Resolved

Description

The linked variable is prefixed with an underscore yet is declared as public .

Recommendation

We advise that the underscore is omitted per the Solidity style guide.

Alleviation

The team opted to consider our references and removed the underscore from the public variable.

SSO-04 | Use Of `memory` Over `storage`

Category	Severity	Location	Status
Optimization	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 35	✓ Resolved

Description

The linked variable is redundantly stored in `storage` , as `storage` look-ups make the gas price higher.

Recommendation

We advise the team to store the result variable in `memory` instead of the `storage`.

Alleviation

The team opted to consider our references and stored the result variable in `memory` instead of the `storage`.

SSO-05 | Potential High `gas` Operation

Category	Severity	Location	Status
Optimization	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 72~74, 105~112	ⓘ Acknowledged

Description

The linked functions iteratively assign values to a `mapping` in `storage`, based on the length of an input array

Recommendation

We advise the team to set upper boundary to the input array length.

Alleviation

The case was a situational and no alleviations were applied.

SSO-06 | Check Against The Zero Address

Category	Severity	Location	Status
Volatile Code	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 31~42, 79~88	✓ Resolved

Description

The linked functions should check the value of their respective `contractAddress` parameter.

Recommendation

We advise the team to add a `require` statement to check against the zero address.

```
require(contractAddress != address(0), "Error Message");
```

Alleviation

The team opted to consider our references and added a `require` statement to check against the zero address, as recommended.

SSO-07 | Different Compiler Versions

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 2	✓ Resolved

Description

If the compiler version is between `0.4.21` and `0.4.26`, then the contract raises a compilation error due to the keyword `payable` .

Recommendation

We advise that compiler versions below `0.5.0` should be avoided

Alleviation

The team opted to consider our references and changed to compiler version `0.5.17`.

SSO-08 | Malicious Hash Collision

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939) : 44~53	✓ Resolved

Description

Since an empty `bytes4` variable, i.e. 0, points to the default score of a contract, it is possible to have the same score applied to a function of the contract as well. The "identifier" of a contract is simply the first 4 bytes of the `keccak256` hash of the signature, meaning that an attacker would simply need to generate a function signature that results in a `keccak256` hash with 4 leading zeroes which is not an impossible achievement.

Recommendation

We advise that the default grade of a contract is either stored in a different data structure or a sanity check is put in place.

Alleviation

The team opted to consider our references and changed the codebase to cover the edge case, as pointed in this exhibit.

SSO-09 | `initialize` Paradigm

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracle.sol (a09e939): 117~121	ⓘ Acknowledged

Description

The `initialize` function of a contract should be invokable only once via sanity checks. Here, it is possible to subsequently call it multiple times.

Recommendation

We advise that a sanity check is imposed whereby the value of `_defaultScore` is ensured to be `0`. Additionally, we would advise a sanity check on the `updateDefaultScore` function that ensures the new `score` is not `0`.

Alleviation

The case was a situational and no alleviations were applied.

SSP-01 | Uncommon Naming Convention

Category	Severity	Location	Status
Coding Style	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracleProxy.sol (a09e939): 8	✓ Resolved

Description

The linked variable is prefixed with an underscore yet is declared as public.

Recommendation

We advise that the underscore is omitted per the Solidity style guide.

Alleviation

The team opted to consider our references and removed the underscore from the public variable.

SSP-02 | Different Compiler Versions

Category	Severity	Location	Status
Language Specific	● Informational	projects/ShentuSecurityOracle/ShentuSecurityOracleProxy.sol (a09e939): 2	✓ Resolved

Description

If the compiler version is above `0.6.0`, then the contract raises a compilation error due to the `fallback()` function in the Proxy.sol file.

Recommendation

We advise that compiler versions above `0.6.0` should be avoided or change the following function of the Proxy.sol file:

```
function() external payable {
_fallback();
}
```

to

```
fallback () external payable {
_fallback();
}
```

Alleviation

The team opted to consider our references and changed to compiler version `0.5.17`.

Appendix

Finding Categories

Volatile Code

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

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 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.