Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Security Token
Website: Eurst.io
Platform: Binance Smart Chain
Language: Solidity
Date: October 23rd, 2021

Table of contents

Introduction 4

Project Background 4

Audit Scope 4

Claimed Smart Contract Features 5

Audit Summary 6

Technical Quick Stats 7

Code Quality 8

Documentation 8

Use of Dependencies 8

AS-IS overview 9

Severity Definitions 12

Audit Findings 13

Conclusion 17

Our Methodology 18

Disclaimers 20

Appendix

- Code Flow Diagram 21
- Slither Results Log 23
- Solidity static analysis 27
- Solhint Linter 34

Email: audit@EtherAuthority.io

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by the Eurst team to perform the Security audit of the Security Token smart contract code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on October 23rd, 2021.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

SecurityTokenSBT is a token contract based on the standard ERC-1594 and ERC-20.

Audit scope

Name	**Code Review and Security Analysis Report for SecurityTokenSBT Smart Contract**
Platform	**BSC / Solidity**
File	SecurityTokenSBT.sol
File 1 MD5 Hash	261D1EF67841481F6F7E5ED574E32A6C
File	UserRegistry.sol
File 2 MD5 Hash	D07AEE80FCB2471F081D9E3BB3700D68
Audit Date	October 23rd, 2021

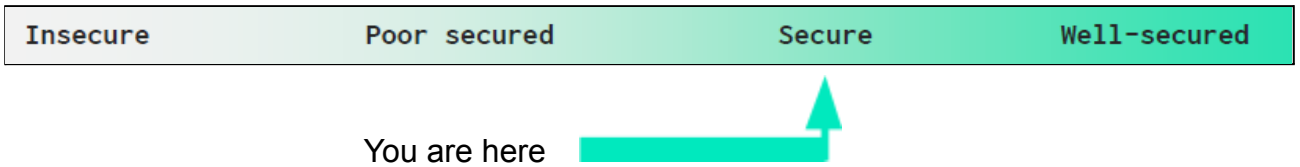
PS: This project contains other libraries and standard code from the openzepelin. Those contracts are assumed to be audited, and thus are not included in this audit scope.

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
• Token name : titan estate • Token symbol : TTS • Decimals : 18 Deployer can set Controller, UserRegistry	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **“Secured”**. This token contract does contain owner control, which does not make it fully decentralized.



We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.
All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 5 low and some very low level issues.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Moderated
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderated
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Moderated
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Moderated
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

This audit scope has 2 smart contract files. Smart contracts contains Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in SecurityTokenSBT Token are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the SecurityTokenSBT Token.

The Eurst team has **not** provided scenario and unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are **not** well commented on smart contracts.

Documentation

We were given a SecurityTokenSBT Token smart contract code in the form of a file.The hashes of that code are mentioned above in the table.

As mentioned above, code parts are **not well** commented. So it is not easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Functions

SecurityTokenSBT.sol

Sl.	Functions	Type	Observation	Conclusion
1	onlyController	modifier	Passed	No Issue
2	constructor	write	Passed	No Issue
3	transferWithData	external	Passed	No Issue
4	transferFromWithData	external	Passed	No Issue
5	issue	external	Passed	No Issue
6	redeem	external	Passed	No Issue
7	redeemFrom	external	Passed	No Issue
8	isControllable	read	Passed	No Issue
9	controllerTransfer	external	access only Controller	No Issue
10	controllerRedeem	external	access only Controller	No Issue
11	setController	external	Function input parameters lack of check	Refer Audit Findings
12	_isControllable	read	Passed	No Issue
13	setUserRegistry	external	Function input parameters lack of check	Refer Audit Findings
14	balanceOfByPartition	read	Passed	No Issue
15	partitionsOf	read	Infinite loop possibility	Refer Audit Findings
16	transferByPartition	external	Missing error message, Unnamed return variable can remain unassigned	Refer Audit Findings
17	canTransferByPartition	read	Unused parameter	Refer Audit Findings
18	_validPartitionForReceiver	read	Infinite loop possibility	Refer Audit Findings
19	isOperator	read	Passed	No Issue
20	isOperatorForPartition	read	Passed	No Issue
21	authorizeOperator	external	Function input parameters lack of check	Refer Audit Findings
22	revokeOperator	external	Function input parameters lack of check	Refer Audit Findings

23	authorizeOperatorByPartition	external	Function input parameters lack of check	Refer Audit Findings
24	revokeOperatorByPartition	external	Passed	No Issue
25	operatorTransferByPartition	external	Unnamed return variable can remain unassigned	Refer Audit Findings
26	issueByPartition	external	access only Owner	No Issue
27	redeemByPartition	external	Passed	No Issue
28	operatorRedeemByPartition	external	Passed	No Issue
29	owner	read	Passed	No Issue
30	onlyOwner	modifier	Passed	No Issue
31	renounceOwnership	write	access only Owner	No Issue
32	transferOwnership	write	access only Owner	No Issue

Userregistry.sol

Sl.	Functions	Type	Observation	Conclusion
1	onlyController	modifier	Passed	No Issue
2	constructor	write	Passed	No Issue
3	set_redeemption_address_count	write	access only Owner	No Issue
4	setToken	write	Critical operation lacks event log	Refer Audit Findings
5	setController	write	access only Owner	No Issue
6	setKyc	write	access only Owner	No Issue
7	setMinBurnBound	write	access only Owner	No Issue
8	setMaxBurnBound	write	access only Owner	No Issue
9	registerNewUser	write	access only Owner	No Issue
10	getUser	read	Passed	No Issue
11	getUserById	read	Passed	No Issue
12	setUserId	write	Critical operation lacks event log	Refer Audit Findings
13	userKycVerified	write	Function input parameters lack of check	Refer Audit Findings
14	userKycUnverified	write	Function input parameters lack of check	Refer Audit Findings
15	enableRedeemAddress	write	Function input parameters lack of check	Refer Audit Findings
16	disableRedeemAddress	write	Function input parameters lack of check	Refer Audit Findings
17	enableIssueAddress	write	access only Owner	No Issue
18	disableIssueAddress	write	access only Owner	No Issue

19	verifyKycEnableRedeem	write	Function input parameters lack of check	Refer Audit Findings
20	unverifyKycDisableRedeem	write	Function input parameters lack of check	Refer Audit Findings
21	blockAccount	write	access only Owner	No Issue
22	unblockAccount	write	access only Owner	No Issue
23	getUserByRedeemAddress	read	Passed	No Issue
24	getRedeemAddress	read	Passed	No Issue
25	canTransfer	external	access only Token	No Issue
26	canTransferFrom	external	access only Token	No Issue
27	canMint	external	access only Token	No Issue
28	canBurn	external	access only Token	No Issue
29	onlyToken	external	access only Token	No Issue
30	setDocument	external	access only Owner	No Issue
31	removeDocument	external	access only Owner	No Issue
32	getDocument	read	Passed	No Issue
33	getAllDocuments	read	Passed	No Issue
34	isControllable	read	Passed	No Issue
35	controllerTransfer	external	access only Controller	No Issue
36	controllerRedeem	external	access only Controller	No Issue
37	finalizeControllable	external	access only Owner	No Issue
38	checkTransferByPartition	read	Function return only true	Refer Audit Findings
39	checkOperatorTransferByPartition	read	Function return only true	Refer Audit Findings
40	checkRevokeOperatorByPartition	read	Function return only true	Refer Audit Findings
41	checkController	read	Function return only true	Refer Audit Findings

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Email: audit@EtherAuthority.io

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

SecurityTokenSBT.sol

(1) Infinite loops possibility:

As array elements will increase, then it will cost more and more gas. And eventually, it will stop all the functionality. After several hundreds of transactions, all those functions depending on it will stop. We suggest avoiding loops. For example, use mapping to store the array index. And query that data directly, instead of looping through all the elements to find an element.

Resolution: Adjust logic to replace loops with mapping or other code structure.

- partitionsOf() - partitions.length
- _validPartitionForReceiver() - partitions.length

(2) Function input parameters lack of check:

Once KYC is verified , without Unverify User is able to verify it again multiple times. Same is in the functions below:

- userKycUnverified
- enableRedeemAddress
- disableRedeemAddress
- verifyKycEnableRedeem
- unverifyKycDisableRedeem

Variable validation is not performed in the functions below :

- setController
- revokeOperator
- authorizeOperatorByPartition
- setUserRegistry

Resolution: We suggest checking for " KYC already verified". This might save some gas. Apply the same for other mentioned functions.
Use some validation like for an integer type check variable should be greater than 0 and for an address type check variable should not be address(0).

UserRegistry.sol

(1) Contracts using the experimental ABIEncoderV2:

Because the ABIEncoderV2 is experimental, it would be risky to release the project using it. Moreover, the recent findings show that it is likely that other important bugs are yet to be found. https://blog.ethereum.org/2019/03/26/solidity-optimizer-and-abiencoderv2-bug

Resolution: We suggest avoiding using this if logically possible.

(2) Function return only true:

These functions return only true regardless of checking the parameters or any condition. These functions are used in SecurityTokenSBT.sol file

```
function checkTransferByPartition(address _from, address _to, bytes32 _partition, uint256 _value) external view returns(bool){
    return true;
}

function checkOperatorTransferByPartition(address _operator, address _tokenHolder, bytes32 _partition) external view returns(bool){
    return true;
}

function checkRevokeOperatorByPartition(address _operator, bytes32 _partition) external view returns(bool){
    return true;
}

function checkController(address _from, address _to) external view returns(bool){
    return true;
}
```

Resolution: This will just cost the gas and do nothing. We suggest to remove those function definitions and usage.

(3) Critical operation lacks event log:

Missing event log for below functions:

- setToken
- setUserId

Resolution: Write an event log for listed events.

Very Low / Informational / Best practices:

SecurityTokenSBT.sol

(1) Warning: SPDX license identifier:

```
Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a
comment containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-
Identifier: UNLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> SecurityTokenSBT.sol
```

Warning: SPDX license identifier not provided in source file.

Resolution: We suggest adding the SPDX license identifier.

(2) Missing error message:

Require without error message.

```
function _transferByPartition(address _from, address _to, uint256 _value, bytes32 _partition, bytes memory _data,
    userRegistry.checkTransferByPartition(_from, _to, _partition, _value);
    require(_validPartition(_partition, _from));
    require(partitions[_from][partitionToIndex[_from][_partition] - 1].amount >= _value, "Insufficient balance");
    require(_to != address(0));
    uint256 _fromIndex = partitionToIndex[_from][_partition] - 1;
```

Resolution: Write an appropriate error message.

(3) Unused parameter:

_data parameter has not been used in the function.

```
/// @return The partition to which the transferred tokens were allocated for the _to address
function canTransferByPartition(address _from, address _to, bytes32 _partition, uint256 _value, bytes calldata _data) external view returns (bytes1, bytes32, bytes32) {
    userRegistry.checkTransferByPartition(_from, _to, _partition, _value);
    if (!_validPartition(_partition, _from))
        return (0x50, "Partition not exists", bytes32(""));
    else if (partitions[_from][partitionToIndex[_from][_partition]].amount < _value)
        return (0x52, "Insufficent balance", bytes32(""));
    else if (_to == address(0))
        return (0x57, "Invalid receiver", bytes32(""));
    // Call function to get the receiver's partition. For current implementation returning the same as sender's
    return (0x51, "Success", _partition);
    }
```

Resolution: Remove the unused parameter.

(4) Unnamed return variable can remain unassigned:

Unnamed return variables can remain unassigned.

```
function transferByPartition(bytes32 _partition, address _to, uint256 _value, bytes calldata _data) external returns (bytes32) {
    _transferByPartition(msg.sender, _to, _value, _partition, _data, address(0), "");

}
```

```
/// @return The partition to which the transferred tokens were allocated for the _to address
function operatorTransferByPartition(bytes32 _partition, address _from, address _to, uint256 _value, bytes calldata _data, bytes calldata _operatorData) external returns (bytes32) {
    userRegistry.checkOperatorTransferByPartition(_from, _to, _partition);
    require(
        isOperator(msg.sender, _from) || isOperatorForPartition(_partition, msg.sender, _from),
        "Not authorised"
    );
    _transferByPartition(_from, _to, _value, _partition, _data, msg.sender, _operatorData);
}
```

Resolution: Add an explicit return with value to all non-reverting code paths or name the variable.

Centralization

These smart contracts have some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

- setController: Owner can set the controller.
- setUserRegistry: Owner can set user registry.
- issueByPartition: Owner can issue tokens by partition.

Conclusion

We were given a contract code. And we have used all possible tests based on given objects as files. We observed some issues in the smart contracts but those issues are not critical ones. So, **it's good to go to production**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured".**

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

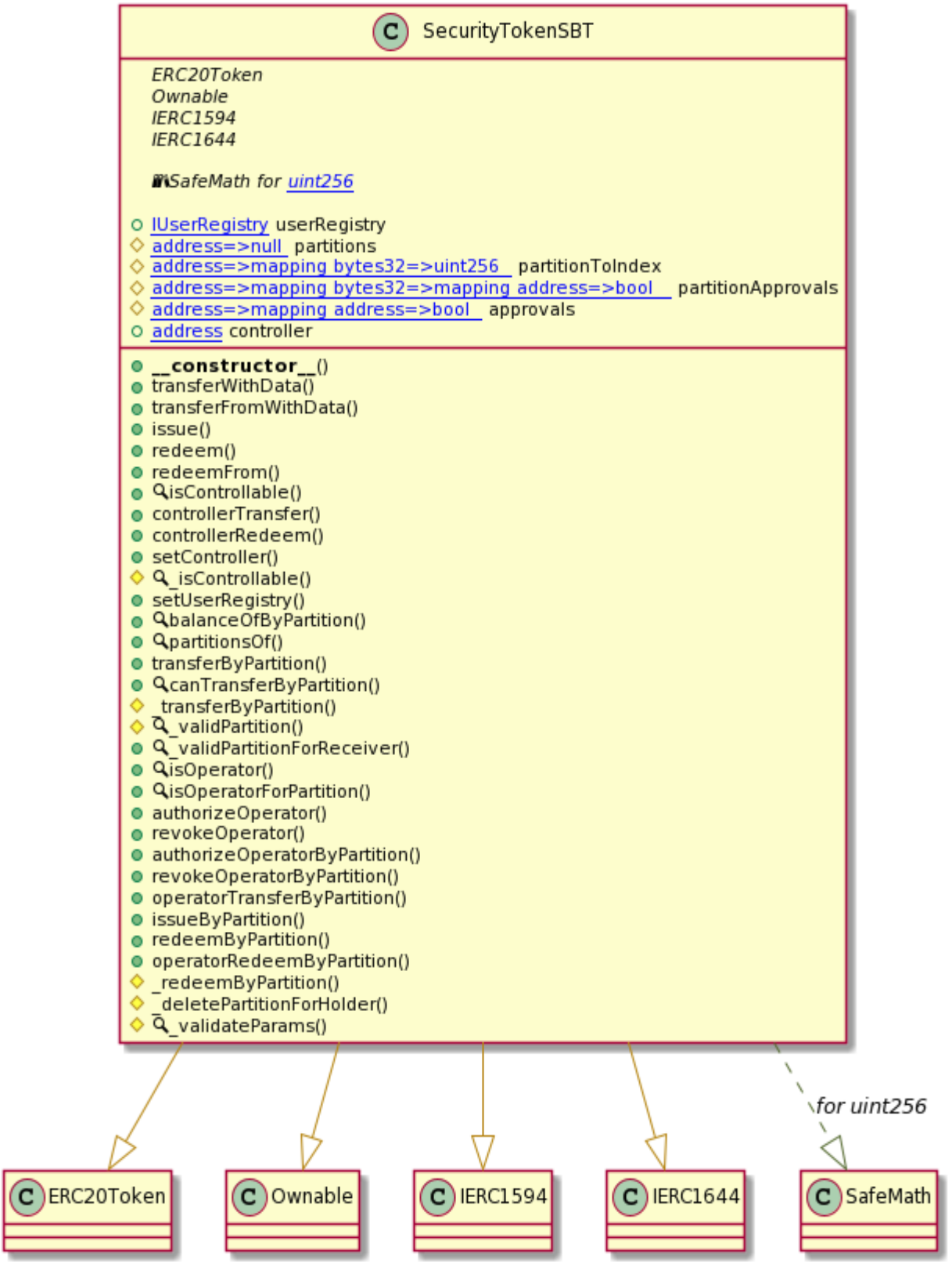
Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can’t guarantee explicit security of the audited smart contracts.

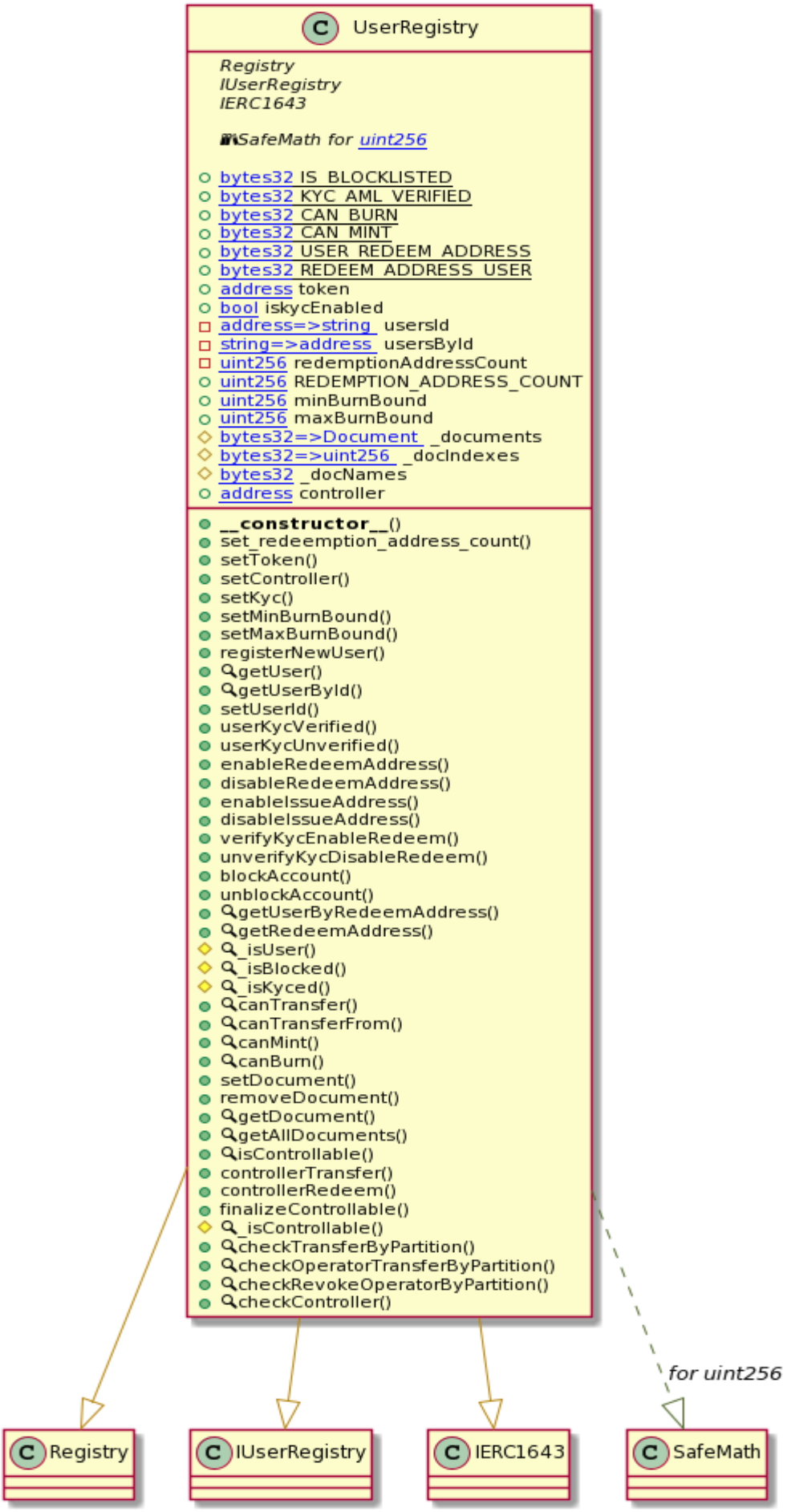
Appendix

Code Flow Diagram - Security Token

SecurityTokenSBT.sol



UserRegistry.sol



Slither Results Log

Slither log >> SecurityTokenSBT.sol

```
root@krestina-desktop:/home/krestina/smartcontract_audits/mycontracts/security-token-main/contracts# slith
er SecurityTokenSBT.sol
Compilation warnings/errors on SecurityTokenSBT.sol:
Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> SecurityTokenSBT.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> ERC1594/IERC1594.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> ERC1644/IERC1644.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> ERC20Token.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> Ownable.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> math/KindMath.sol
```

```
SecurityTokenSBT.transferWithData(address,uint256,bytes) (SecurityTokenSBT.sol#83-86) ignores return value
 by userRegistry.canTransfer(msg.sender,balanceOf(msg.sender),_to,_value,_data) (SecurityTokenSBT.sol#84)
SecurityTokenSBT.transferFromWithData(address,address,uint256,bytes) (SecurityTokenSBT.sol#101-104) ignore
s return value by userRegistry.canTransferFrom(_from,balanceOf(_from),_value,_data,_to,balanceOf(_to)) (Se
curityTokenSBT.sol#102)
SecurityTokenSBT.controllerTransfer(address,address,uint256,bytes,bytes) (SecurityTokenSBT.sol#161-165) ig
nores return value by userRegistry.checkController(_from,_to) (SecurityTokenSBT.sol#162)
SecurityTokenSBT.controllerRedeem(address,uint256,bytes,bytes) (SecurityTokenSBT.sol#168-173) ignores retu
rn value by userRegistry.checkController(_tokenHolder,address(0)) (SecurityTokenSBT.sol#169)
SecurityTokenSBT.balanceOfByPartition(bytes32,address) (SecurityTokenSBT.sol#205-211) ignores return value
 by userRegistry.checkTransferByPartition(_tokenHolder,address(0),_partition,0) (SecurityTokenSBT.sol#206)
SecurityTokenSBT.partitionsOf(address) (SecurityTokenSBT.sol#217-224) ignores return value by userRegistry
.checkTransferByPartition(_tokenHolder,address(0),,0) (SecurityTokenSBT.sol#218)
SecurityTokenSBT.canTransferByPartition(address,address,bytes32,uint256,bytes) (SecurityTokenSBT.sol#248-2
58) ignores return value by userRegistry.checkTransferByPartition(_from,_to,_partition,_value) (SecurityTo
kenSBT.sol#249)
SecurityTokenSBT._transferByPartition(address,address,uint256,bytes32,bytes,address,bytes) (SecurityTokenS
BT.sol#260-279) ignores return value by userRegistry.checkTransferByPartition(_from,_to,_partition,_value)
 (SecurityTokenSBT.sol#261)
SecurityTokenSBT.authorizeOperator(address) (SecurityTokenSBT.sol#317-321) ignores return value by userReg
istry.checkOperatorTransferByPartition(_operator,address(0),) (SecurityTokenSBT.sol#318)
SecurityTokenSBT.revokeOperator(address) (SecurityTokenSBT.sol#326-330) ignores return value by userRegist
ry.checkRevokeOperatorByPartition(_operator,) (SecurityTokenSBT.sol#327)
SecurityTokenSBT.authorizeOperatorByPartition(bytes32,address) (SecurityTokenSBT.sol#335-339) ignores retu
rn value by userRegistry.checkOperatorTransferByPartition(_operator,address(0),_partition) (SecurityTokenS
BT.sol#336)
SecurityTokenSBT.revokeOperatorByPartition(bytes32,address) (SecurityTokenSBT.sol#345-349) ignores return
value by userRegistry.checkRevokeOperatorByPartition(_operator,_partition) (SecurityTokenSBT.sol#346)
SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes) (SecurityTokenSB
T.sol#359-366) ignores return value by userRegistry.checkOperatorTransferByPartition(_from,_to,_partition)
 (SecurityTokenSBT.sol#360)
```

```
KindMath.checkAdd(uint256,uint256) (math/KindMath.sol#41-47) is never used and should be removed
KindMath.checkMul(uint256,uint256) (math/KindMath.sol#13-26) is never used and should be removed
KindMath.checkSub(uint256,uint256) (math/KindMath.sol#31-36) is never used and should be removed
SafeMath.div(uint256,uint256) (math/safeMath.sol#25-32) is never used and should be removed
SafeMath.mod(uint256,uint256) (math/safeMath.sol#58-61) is never used and should be removed
SafeMath.mul(uint256,uint256) (math/safeMath.sol#8-20) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (ERC1594/IERC1594.sol#1) necessitates a version too recent to be trusted. Consider de
ploying with 0.6.12/0.7.6
Pragma version^0.8.9 (ERC1644/IERC1644.sol#1) necessitates a version too recent to be trusted. Consider de
ploying with 0.6.12/0.7.6
Pragma version^0.8.9 (ERC20Token.sol#1) necessitates a version too recent to be trusted. Consider deployin
g with 0.6.12/0.7.6
Pragma version^0.8.0 (Ownable.sol#1) necessitates a version too recent to be trusted. Consider deploying w
ith 0.6.12/0.7.6
Pragma version^0.8.9 (SecurityTokenSBT.sol#1) necessitates a version too recent to be trusted. Consider de
ploying with 0.6.12/0.7.6
Pragma version^0.8.9 (interfaces/IUserRegistry.sol#3) necessitates a version too recent to be trusted. Con
sider deploying with 0.6.12/0.7.6
Pragma version^0.8.9 (math/KindMath.sol#1) necessitates a version too recent to be trusted. Consider deplo
ying with 0.6.12/0.7.6
Pragma version^0.8.9 (math/safeMath.sol#1) necessitates a version too recent to be trusted. Consider deplo
ying with 0.6.12/0.7.6
Pragma version^0.8.0 (openzeppelin-solidity/contracts/token/ERC20/IERC20.sol#3) necessitates a version too
 recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Variable ERC20Token._balances (ERC20Token.sol#10) is not in mixedCase
Variable ERC20Token._allowed (ERC20Token.sol#12) is not in mixedCase
Parameter SecurityTokenSBT.transferWithData(address,uint256,bytes)._to (SecurityTokenSBT.sol#83) is not in
 mixedCase
Parameter SecurityTokenSBT.transferWithData(address,uint256,bytes)._value (SecurityTokenSBT.sol#83) is not
 in mixedCase
Parameter SecurityTokenSBT.transferWithData(address,uint256,bytes)._data (SecurityTokenSBT.sol#83) is not
in mixedCase
```

```
Parameter SecurityTokenSBT.redeem(uint256,bytes)._value (SecurityTokenSBT.sol#128) is not in mixedCase
Parameter SecurityTokenSBT.redeem(uint256,bytes)._data (SecurityTokenSBT.sol#128) is not in mixedCase
Parameter SecurityTokenSBT.redeemFrom(address,uint256,bytes)._tokenHolder (SecurityTokenSBT.sol#143) is no
t in mixedCase
Parameter SecurityTokenSBT.redeemFrom(address,uint256,bytes)._value (SecurityTokenSBT.sol#143) is not in m
ixedCase
Parameter SecurityTokenSBT.redeemFrom(address,uint256,bytes)._data (SecurityTokenSBT.sol#143) is not in mi
xedCase
Parameter SecurityTokenSBT.controllerTransfer(address,address,uint256,bytes,bytes)._from (SecurityTokenSBT
.sol#161) is not in mixedCase
Parameter SecurityTokenSBT.controllerTransfer(address,address,uint256,bytes,bytes)._to (SecurityTokenSBT.s
ol#161) is not in mixedCase
Parameter SecurityTokenSBT.controllerTransfer(address,address,uint256,bytes,bytes)._value (SecurityTokenSB
T.sol#161) is not in mixedCase
Parameter SecurityTokenSBT.controllerTransfer(address,address,uint256,bytes,bytes)._data (SecurityTokenSBT
.sol#161) is not in mixedCase
Parameter SecurityTokenSBT.controllerTransfer(address,address,uint256,bytes,bytes)._operatorData (Security
TokenSBT.sol#161) is not in mixedCase
Parameter SecurityTokenSBT.controllerRedeem(address,uint256,bytes,bytes)._tokenHolder (SecurityTokenSBT.so
l#168) is not in mixedCase
Parameter SecurityTokenSBT.controllerRedeem(address,uint256,bytes,bytes)._value (SecurityTokenSBT.sol#168)
 is not in mixedCase
Parameter SecurityTokenSBT.controllerRedeem(address,uint256,bytes,bytes)._data (SecurityTokenSBT.sol#168)
is not in mixedCase
Parameter SecurityTokenSBT.controllerRedeem(address,uint256,bytes,bytes)._operatorData (SecurityTokenSBT.s
ol#168) is not in mixedCase
Parameter SecurityTokenSBT.setController(address)._controller (SecurityTokenSBT.sol#176) is not in mixedCa
se
Parameter SecurityTokenSBT.setUserRegistry(IUserRegistry)._userRegistry (SecurityTokenSBT.sol#192) is not
in mixedCase
Parameter SecurityTokenSBT.balanceOfByPartition(bytes32,address)._partition (SecurityTokenSBT.sol#205) is
not in mixedCase
Parameter SecurityTokenSBT.balanceOfByPartition(bytes32,address)._tokenHolder (SecurityTokenSBT.sol#205) i
s not in mixedCase
Parameter SecurityTokenSBT.partitionsOf(address)._tokenHolder (SecurityTokenSBT.sol#217) is not in mixedCa
se
Parameter SecurityTokenSBT.transferByPartition(bytes32,address,uint256,bytes)._partition (SecurityTokenSBT
.sol#233) is not in mixedCase
```

```
#311) is not in mixedCase
Parameter SecurityTokenSBT.isOperatorForPartition(bytes32,address,address)._tokenHolder (SecurityTokenSBT.
sol#311) is not in mixedCase
Parameter SecurityTokenSBT.authorizeOperator(address)._operator (SecurityTokenSBT.sol#317) is not in mixed
Case
Parameter SecurityTokenSBT.revokeOperator(address)._operator (SecurityTokenSBT.sol#326) is not in mixedCas
e
Parameter SecurityTokenSBT.authorizeOperatorByPartition(bytes32,address)._partition (SecurityTokenSBT.sol#
335) is not in mixedCase
Parameter SecurityTokenSBT.authorizeOperatorByPartition(bytes32,address)._operator (SecurityTokenSBT.sol#3
35) is not in mixedCase
Parameter SecurityTokenSBT.revokeOperatorByPartition(bytes32,address)._partition (SecurityTokenSBT.sol#345
) is not in mixedCase
Parameter SecurityTokenSBT.revokeOperatorByPartition(bytes32,address)._operator (SecurityTokenSBT.sol#345)
 is not in mixedCase
Parameter SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes)._partit
ion (SecurityTokenSBT.sol#359) is not in mixedCase
Parameter SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes)._from
(SecurityTokenSBT.sol#359) is not in mixedCase
Parameter SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes)._to (S
ecurityTokenSBT.sol#359) is not in mixedCase
Parameter SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes)._value
 (SecurityTokenSBT.sol#359) is not in mixedCase
Parameter SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes)._data
(SecurityTokenSBT.sol#359) is not in mixedCase
Parameter SecurityTokenSBT.operatorTransferByPartition(bytes32,address,address,uint256,bytes,bytes)._opera
torData (SecurityTokenSBT.sol#359) is not in mixedCase
Parameter SecurityTokenSBT.issueByPartition(bytes32,address,uint256,bytes)._partition (SecurityTokenSBT.so
l#374) is not in mixedCase
Parameter SecurityTokenSBT.issueByPartition(bytes32,address,uint256,bytes)._tokenHolder (SecurityTokenSBT.
sol#374) is not in mixedCase
Parameter SecurityTokenSBT.issueByPartition(bytes32,address,uint256,bytes)._value (SecurityTokenSBT.sol#37
4) is not in mixedCase
Parameter SecurityTokenSBT.issueByPartition(bytes32,address,uint256,bytes)._data (SecurityTokenSBT.sol#374
) is not in mixedCase
Parameter SecurityTokenSBT.redeemByPartition(bytes32,uint256,bytes)._partition (SecurityTokenSBT.sol#396)
is not in mixedCase
Parameter SecurityTokenSBT.redeemByPartition(bytes32,uint256,bytes)._value (SecurityTokenSBT.sol#396) is n
```

```
Parameter SecurityTokenSBT.operatorRedeemByPartition(bytes32,address,uint256,bytes,bytes)._partition (Secu
rityTokenSBT.sol#401) is not in mixedCase
Parameter SecurityTokenSBT.operatorRedeemByPartition(bytes32,address,uint256,bytes,bytes)._tokenHolder (Se
curityTokenSBT.sol#401) is not in mixedCase
Parameter SecurityTokenSBT.operatorRedeemByPartition(bytes32,address,uint256,bytes,bytes)._value (Security
TokenSBT.sol#401) is not in mixedCase
Parameter SecurityTokenSBT.operatorRedeemByPartition(bytes32,address,uint256,bytes,bytes)._data (SecurityT
okenSBT.sol#401) is not in mixedCase
Parameter SecurityTokenSBT.operatorRedeemByPartition(bytes32,address,uint256,bytes,bytes)._operatorData (S
ecurityTokenSBT.sol#401) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-co
nventions

totalSupply() should be declared external:
        - ERC20Token.totalSupply() (ERC20Token.sol#19-21)
allowance(address,address) should be declared external:
        - ERC20Token.allowance(address,address) (ERC20Token.sol#38-47)
approve(address,uint256) should be declared external:
        - ERC20Token.approve(address,uint256) (ERC20Token.sol#58-64)
transfer(address,uint256) should be declared external:
        - ERC20Token.transfer(address,uint256) (ERC20Token.sol#71-74)
transferFrom(address,address,uint256) should be declared external:
        - ERC20Token.transferFrom(address,address,uint256) (ERC20Token.sol#82-92)
increaseAllowance(address,uint256) should be declared external:
        - ERC20Token.increaseAllowance(address,uint256) (ERC20Token.sol#117-130)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20Token.decreaseAllowance(address,uint256) (ERC20Token.sol#141-154)
owner() should be declared external:
        - Ownable.owner() (Ownable.sol#25-27)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Ownable.sol#50-53)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Ownable.sol#59-61)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-dec
lared-external
```

Slither log >> UserRegistry.sol

```
root@krestina-desktop:/home/krestina/smartcontract_audits/mycontracts/security-token-main/contracts#
root@krestina-desktop:/home/krestina/smartcontract_audits/mycontracts/security-token-main/contracts# solc
--version
solc, the solidity compiler commandline interface
Version: 0.8.9+commit.e5eed63a.Linux.g++
root@krestina-desktop:/home/krestina/smartcontract_audits/mycontracts/security-token-main/contracts# slith
er UserRegistry.sol
Compilation warnings/errors on UserRegistry.sol:
Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> ERC1643/IERC1643.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> ERC1644/IERC1644.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> math/safeMath.sol

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment
 containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: U
NLICENSED" for non-open-source code. Please see https://spdx.org for more information.
--> Ownable.sol

Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "
abstract" is sufficient.
   --> UserRegistry.sol:101:5:
    |
101 |     constructor(
    |     ^ (Relevant source part starts here and spans across multiple lines).

Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> UserRegistry.sol:448:93:
```

```
Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> UserRegistry.sol:658:87:
    |
658 |     function checkTransferByPartition(address _from, address _to, bytes32 _partition, uint256 _value
) external view returns(bool){
    |                                                                                       ^^^^^^^^^^^^^^

Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> UserRegistry.sol:662:47:
    |
662 |     function checkOperatorTransferByPartition(address _operator, address _tokenHolder, bytes32 _part
ition) external view returns(bool){
    |                                               ^^^^^^^^^^^^^^^^

Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> UserRegistry.sol:662:66:
    |
662 |     function checkOperatorTransferByPartition(address _operator, address _tokenHolder, bytes32 _part
ition) external view returns(bool){
    |                                                                  ^^^^^^^^^^^^^^^^^^^

Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> UserRegistry.sol:662:88:
    |
662 |     function checkOperatorTransferByPartition(address _operator, address _tokenHolder, bytes32 _part
ition) external view returns(bool){
    |                                                                                       ^^^^^^^^^^^^
^^^^^

Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> UserRegistry.sol:666:45:
```

```
Different versions of Solidity is used:
        - Version used: ['^0.8.0', '^0.8.9']
        - ^0.8.9 (ERC1643/IERC1643.sol#1)
        - ^0.8.9 (ERC1644/IERC1644.sol#1)
        - ^0.8.0 (Ownable.sol#1)
        - ^0.8.9 (Registry.sol#2)
        - ABIEncoderV2 (Registry.sol#3)
        - ^0.8.9 (UserRegistry.sol#2)
        - ABIEncoderV2 (UserRegistry.sol#3)
        - ^0.8.9 (interfaces/IUserRegistry.sol#3)
        - ^0.8.9 (math/safeMath.sol#1)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-u
sed

SafeMath.add(uint256,uint256) (math/safeMath.sol#47-52) is never used and should be removed
SafeMath.div(uint256,uint256) (math/safeMath.sol#25-32) is never used and should be removed
SafeMath.mod(uint256,uint256) (math/safeMath.sol#58-61) is never used and should be removed
SafeMath.mul(uint256,uint256) (math/safeMath.sol#8-20) is never used and should be removed
SafeMath.sub(uint256,uint256) (math/safeMath.sol#37-42) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.9 (ERC1643/IERC1643.sol#1) necessitates a version too recent to be trusted. Consider de
ploying with 0.6.12/0.7.6
Pragma version^0.8.9 (ERC1644/IERC1644.sol#1) necessitates a version too recent to be trusted. Consider de
ploying with 0.6.12/0.7.6
Pragma version^0.8.0 (Ownable.sol#1) necessitates a version too recent to be trusted. Consider deploying w
ith 0.6.12/0.7.6
Pragma version^0.8.9 (Registry.sol#2) necessitates a version too recent to be trusted. Consider deploying
with 0.6.12/0.7.6
Pragma version^0.8.9 (UserRegistry.sol#2) necessitates a version too recent to be trusted. Consider deploy
ing with 0.6.12/0.7.6
Pragma version^0.8.9 (interfaces/IUserRegistry.sol#3) necessitates a version too recent to be trusted. Con
sider deploying with 0.6.12/0.7.6
Pragma version^0.8.9 (math/safeMath.sol#1) necessitates a version too recent to be trusted. Consider deplo
ying with 0.6.12/0.7.6
solc-0.8.9 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

```
setKyc(bool) should be declared external:
        - UserRegistry.setKyc(bool) (UserRegistry.sol#126-128)
setMinBurnBound(uint256) should be declared external:
        - UserRegistry.setMinBurnBound(uint256) (UserRegistry.sol#130-135)
setMaxBurnBound(uint256) should be declared external:
        - UserRegistry.setMaxBurnBound(uint256) (UserRegistry.sol#137-142)
registerNewUser(address,string) should be declared external:
        - UserRegistry.registerNewUser(address,string) (UserRegistry.sol#144-169)
getUserById(string) should be declared external:
        - UserRegistry.getUserById(string) (UserRegistry.sol#199-206)
setUserId(address,string) should be declared external:
        - UserRegistry.setUserId(address,string) (UserRegistry.sol#217-224)
userKycVerified(address) should be declared external:
        - UserRegistry.userKycVerified(address) (UserRegistry.sol#235-241)
userKycUnverified(address) should be declared external:
        - UserRegistry.userKycUnverified(address) (UserRegistry.sol#252-258)
enableRedeemAddress(address) should be declared external:
        - UserRegistry.enableRedeemAddress(address) (UserRegistry.sol#270-277)
disableRedeemAddress(address) should be declared external:
        - UserRegistry.disableRedeemAddress(address) (UserRegistry.sol#288-294)
enableIssueAddress(address) should be declared external:
        - UserRegistry.enableIssueAddress(address) (UserRegistry.sol#306-313)
disableIssueAddress(address) should be declared external:
        - UserRegistry.disableIssueAddress(address) (UserRegistry.sol#324-330)
verifyKycEnableRedeem(address) should be declared external:
        - UserRegistry.verifyKycEnableRedeem(address) (UserRegistry.sol#343-351)
unverifyKycDisableRedeem(address) should be declared external:
        - UserRegistry.unverifyKycDisableRedeem(address) (UserRegistry.sol#364-372)
blockAccount(address) should be declared external:
        - UserRegistry.blockAccount(address) (UserRegistry.sol#383-388)
unblockAccount(address) should be declared external:
        - UserRegistry.unblockAccount(address) (UserRegistry.sol#399-404)
getUserByRedeemAddress(address) should be declared external:
        - UserRegistry.getUserByRedeemAddress(address) (UserRegistry.sol#409-415)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-dec
lared-external
```

Solidity Static Analysis

SecurityTokenSBT.sol

Security

Check-effects-interaction:
Potential violation of Checks-Effects-Interaction pattern in SecurityTokenSBT.transferWithData(address,uint256,bytes): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 620:4:

Check-effects-interaction:
Potential violation of Checks-Effects-Interaction pattern in SecurityTokenSBT.transferFromWithData(address,address,uint256,bytes): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 638:4:

Check-effects-interaction:
Potential violation of Checks-Effects-Interaction pattern in SecurityTokenSBT.issue(address,uint256,bytes): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 652:4:

Check-effects-interaction:
Potential violation of Checks-Effects-Interaction pattern in SecurityTokenSBT.redeem(uint256,bytes): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 665:4:

Check-effects-interaction:
Potential violation of Checks-Effects-Interaction pattern in SecurityTokenSBT.redeemFrom(address,uint256,bytes): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 680:4:

Gas & Economy

Gas costs:
Gas requirement of function ERC20Token.allowance is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 285:4:

Gas costs:
Gas requirement of function SecurityTokenSBT.allowance is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 285:4:

Gas costs:
Gas requirement of function ERC20Token.approve is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 305:4:

Gas costs:
Gas requirement of function SecurityTokenSBT.approve is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 305:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.

more

Pos: 757:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.

more

Pos: 827:8:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.

more

Pos: 966:8:

Miscellaneous

Constant/View/Pure functions:

IERC1594.transferWithData(address,uint256,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 7:4:

Constant/View/Pure functions:

IERC1594.transferFromWithData(address,address,uint256,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 8:4:

Constant/View/Pure functions:

IERC1594.issue(address,uint256,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 12:4:

Constant/View/Pure functions:

IERC1594.redeem(uint256,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 15:4:

Similar variable names:

SecurityTokenSBT.balanceOfByPartition(bytes32,address) : Variables have very similar names "partitions" and "_partition". Note: Modifiers are currently not considered by this static analysis.
Pos: 744:28:

Similar variable names:

SecurityTokenSBT.balanceOfByPartition(bytes32,address) : Variables have very similar names "partitions" and "_partition". Note: Modifiers are currently not considered by this static analysis.
Pos: 745:19:

Similar variable names:

SecurityTokenSBT.balanceOfByPartition(bytes32,address) : Variables have very similar names "partitions" and "_partition". Note: Modifiers are currently not considered by this static analysis.
Pos: 745:75:

Similar variable names:

SecurityTokenSBT.transferByPartition(bytes32,address,uint256,bytes) : Variables have very similar names "partitions" and "_partition". Note: Modifiers are currently not considered by this static analysis.
Pos: 771:54:

Similar variable names:

SecurityTokenSBT.canTransferByPartition(address,address,bytes32,uint256,bytes) : Variables have very similar names "partitions" and "_partition". Note: Modifiers are currently not considered by this static analysis.
Pos: 786:58:

Similar variable names:

SecurityTokenSBT.canTransferByPartition(address,address,bytes32,uint256,bytes) : Variables have very similar names "partitions" and "_partition". Note: Modifiers are currently not considered by this static analysis.
Pos: 787:29:

No return:

IERC1644.isControllable(): Defines a return type but never explicitly returns a value.
Pos: 30:4:

No return:

IERC20.totalSupply(): Defines a return type but never explicitly returns a value.
Pos: 124:4:

No return:

IERC20.balanceOf(address): Defines a return type but never explicitly returns a value.
Pos: 129:4:

No return:

IERC20.transfer(address,uint256): Defines a return type but never explicitly returns a value.
Pos: 138:4:

No return:

IERC20.allowance(address,address): Defines a return type but never explicitly returns a value.
Pos: 147:4:

No return:

IERC20.approve(address,uint256): Defines a return type but never explicitly returns a value.
Pos: 163:4:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 914:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 940:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 941:8:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 950:8:

Delete from dynamic array:

Using "delete" on an array leaves a gap. The length of the array remains the same. If you want to remove the empty position you need to shift items manually and update the "length" property.
more
Pos: 970:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 207:16:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 218:20:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 520:12:

UserRegistry.sol

Security

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 95:12:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree.
That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.
more
Pos: 798:52:

Gas & Economy

Gas costs:

Gas requirement of function Registry.setAttribute is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 87:4:

Gas costs:

Gas requirement of function UserRegistry.setAttribute is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed.
Please avoid loops in your functions or actions that modify large areas of storage
(this includes clearing or copying arrays in storage)
Pos: 87:4:

For loop over dynamic array:

Loops that do not have a fixed number of iterations, for example, loops that depend on storage values, have to be used carefully. Due to the block gas limit, transactions can only consume a certain amount of gas. The number of iterations in a loop can grow beyond the block gas limit which can cause the complete contract to be stalled at a certain point. Additionally, using unbounded loops incurs in a lot of avoidable gas costs. Carefully test how many items at maximum you can pass to such functions to make it successful.
more
Pos: 811:8:

Miscellaneous

Constant/View/Pure functions:

IUserRegistry.controllerTransfer(address,address,uint256,bytes,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 142:4:

Constant/View/Pure functions:

IUserRegistry.controllerRedeem(address,uint256,bytes,bytes) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 144:4:

Constant/View/Pure functions:

IERC1643.setDocument(bytes32,string,bytes32) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 170:4:

Constant/View/Pure functions:

IERC1643.removeDocument(bytes32) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 171:4:

Similar variable names:

Registry.setAttribute(address,bytes32,uint256) : Variables have very similar names "attributes" and "_attribute". Note: Modifiers are currently not considered by this static analysis.
Pos: 92:8:

Similar variable names:

Registry.setAttribute(address,bytes32,uint256) : Variables have very similar names "attributes" and "_attribute". Note: Modifiers are currently not considered by this static analysis.
Pos: 92:25:

Similar variable names:

Registry.setAttribute(address,bytes32,uint256) : Variables have very similar names "attributes" and "_attribute". Note: Modifiers are currently not considered by this static analysis.
Pos: 97:32:

Similar variable names:

Registry.hasAttribute(address,bytes32) : Variables have very similar names "attributes" and "_attribute". Note: Modifiers are currently not considered by this static analysis.
Pos: 105:15:

Position in

Similar variable names:

Registry.hasAttribute(address,bytes32) : Variables have very similar names "attributes" and "_attribute". Note: Modifiers are currently not considered by this static analysis.
Pos: 105:32:

Guard conditions:

Use "assert(x)" if you never ever want x to be false, not in any circumstance (apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.
more
Pos: 895:8:

Delete from dynamic array:

Using "delete" on an array leaves a gap. The length of the array remains the same. If you want to remove the empty position you need to shift items manually and update the "length" property.
more
Pos: 477:8:

Delete from dynamic array:

Using "delete" on an array leaves a gap. The length of the array remains the same. If you want to remove the empty position you need to shift items manually and update the "length" property.
more
Pos: 818:8:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 219:16:

Data truncated:

Division of integer values yields an integer value again. That means e.g. 10 / 100 = 0 instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.
Pos: 230:20:

Solhint Linter

SecurityTokenSBT.sol

```
SecurityTokenSBT.sol:2:1: Error: Compiler version ^0.8.9 does not
satisfy the r semver requirement
SecurityTokenSBT.sol:62:5: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
SecurityTokenSBT.sol:558:5: Error: Explicitly mark visibility of
state
SecurityTokenSBT.sol:562:5: Error: Explicitly mark visibility of
state
SecurityTokenSBT.sol:565:5: Error: Explicitly mark visibility of
state
SecurityTokenSBT.sol:568:5: Error: Explicitly mark visibility of
state
SecurityTokenSBT.sol:785:101: Error: Variable "_data" is unused
```

UserRegistry.sol

```
UserRegistry.sol:2:1: Error: Compiler version ^0.8.9 does not satisfy
the r semver requirement
UserRegistry.sol:14:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
UserRegistry.sol:95:13: Error: Avoid to make time-based decisions in
your business logic
UserRegistry.sol:287:20: Error: Variable name must be in mixedCase
UserRegistry.sol:313:5: Error: Explicitly mark visibility of state
UserRegistry.sol:359:9: Error: Variable name must be in mixedCase
UserRegistry.sol:368:5: Error: Function name must be in mixedCase
UserRegistry.sol:368:44: Error: Variable name must be in mixedCase
UserRegistry.sol:702:93: Error: Variable "_data" is unused
UserRegistry.sol:733:9: Error: Variable "_data" is unused
UserRegistry.sol:735:9: Error: Variable "balanceOfTo" is unused
UserRegistry.sol:798:53: Error: Avoid to make time-based decisions in
your business logic
UserRegistry.sol:912:39: Error: Variable "_from" is unused
UserRegistry.sol:912:54: Error: Variable "_to" is unused
UserRegistry.sol:912:67: Error: Variable "_partition" is unused
UserRegistry.sol:912:87: Error: Variable "_value" is unused
UserRegistry.sol:916:47: Error: Variable "_operator" is unused
UserRegistry.sol:916:66: Error: Variable "_tokenHolder" is unused
UserRegistry.sol:916:88: Error: Variable "_partition" is unused
UserRegistry.sol:920:45: Error: Variable "_operator" is unused
UserRegistry.sol:920:64: Error: Variable "_partition" is unused
UserRegistry.sol:924:30: Error: Variable "_from" is unused
UserRegistry.sol:924:45: Error: Variable "_to" is unused
```

Software analysis result:

These software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

Email: audit@EtherAuthority.io