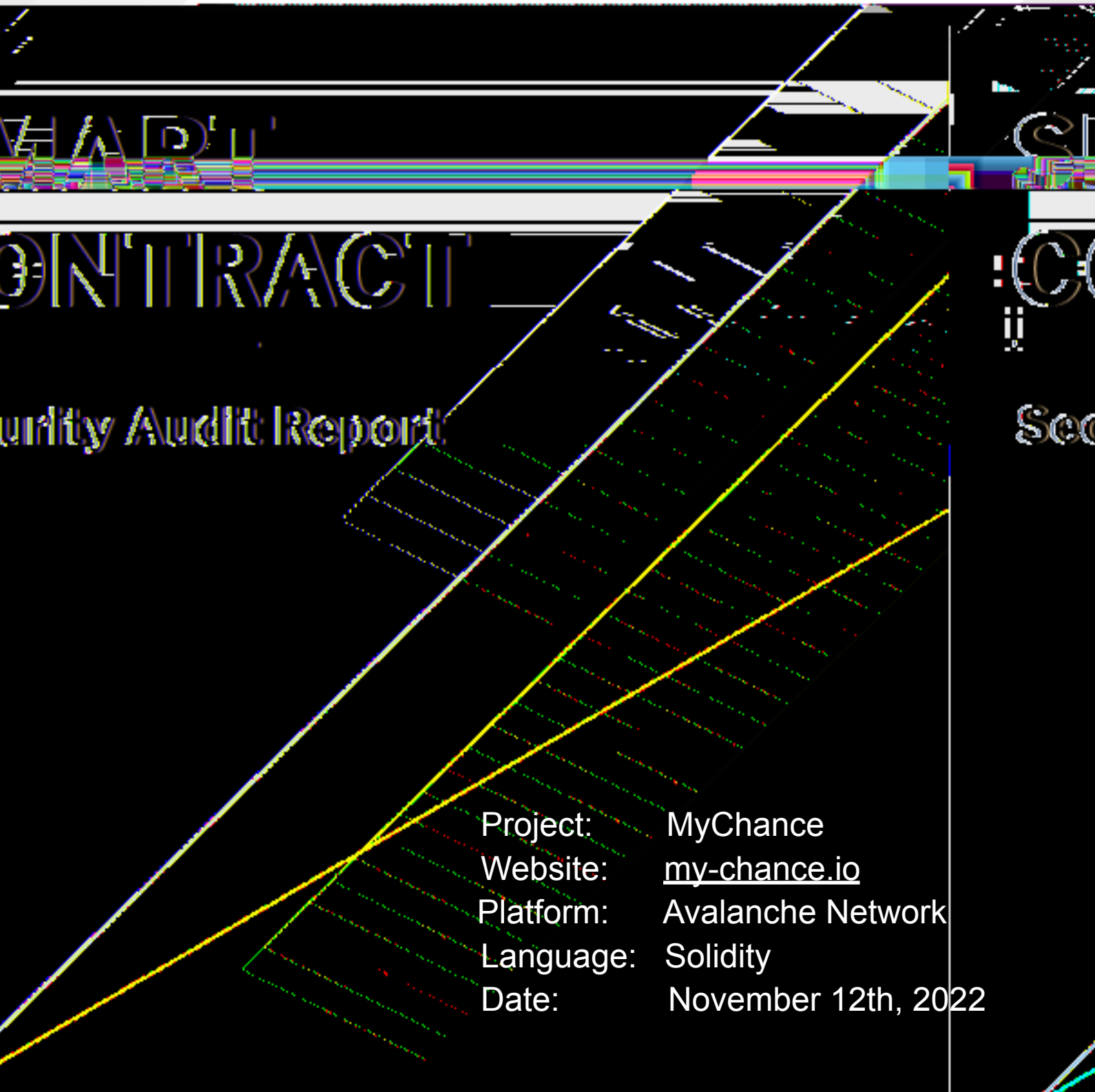
Ether Authority

www.EtherAuthority.io
audit@etherauthority.io



Project: MyChance
Website: my-chance.io
Platform: Avalanche Network
Language: Solidity
Date: November 12th, 2022

Table of contents

Introduction 4
Project Background 4
Audit Scope 5
Claimed Smart Contract Features 6
Audit Summary 7
Technical Quick Stats 8
Code Quality 9
Documentation 9
Use of Dependencies 9
AS-IS overview 10
Severity Definitions 14
Audit Findings 15
Conclusion 19
Our Methodology 20
Disclaimers 22
Appendix

- Code Flow Diagram 23
- Slither Results Log 28
- Solidity static analysis 33
- Solhint Linter 43

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by MyChance to perform the Security audit of the MyChance protocol smart contracts 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 November 12th, 2022.

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

- MyChance is a Lottery protocol which uses priceBonds as an NFT smart contract, having functions like safeMint, safeBurn, pause, unpause, grantRole, revokeRole, renounceRole, supportsInterface, startMigration, migrateMyself, etc. Users can acquire Prize Bonds (which are NFTs) by paying in USDT, DAI or USC, and automatically participate in lotteries. A part of their earnings is shared with charities.
- A random winner chosen by every 7 days' draw wins the total interests generated in the last week.
- These contracts inherit IERC20, Pausable, ERC721Enumerable, AccessControl, ERC721, Ownable, Counters standard smart contracts from the OpenZeppelin library. And KeeperCompatible, LinkTokenInterface, VRFConsumerBaseV2, LinkTokenInterface, ConfirmedOwner, VRFCoordinatorV2Interface standard smart contracts from the chain link library.
- These OpenZeppelin and chain link contracts are considered community audited and time tested, and hence are not part of the audit scope.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Audit scope

Name	Code Review and Security Analysis Report for MyChance Protocol Smart Contracts
Platform	**Avalanche / Solidity**
File 1	MyChance.sol
File 1 MD5 Hash	4A0252C003CD151B7373B5E0A881CA4B
Updated File 1 MD5 Hash	0C27E0475C8F9CFF2C1E50CAAA82C2BF
File 2	Charities.sol
File 2 MD5 Hash	A49286FEAF368714031BDF214DD65391
File 3	PrizeBond.sol
File 3 MD5 Hash	AD5BC9DEA3A1EA58AB7669FC621D8699
Updated File 3 MD5 Hash	B2E4531EDD29E9D546F21A2BF9DA28E5
File 4	RandomRequester.sol
File 4 MD5 Hash	91020DA4F056EB4C053C202D2B0D5BEA
File 5	Roles.sol
File 5 MD5 Hash	5CA0956D90B633BF9E71F71101585C76
Updated File 5 MD5 Hash	03E610EFA5B4C10B8D65A193F490EF5F
Audit Date	November 12th, 2022
Revised Audit Date	November 25th, 2022

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 MyChance.sol • Owner can handle pausable activities. • Main contract that holds the lottery business rules as well as numerous administrative functions.	**YES, This is valid.**
File 2 Charities.sol • Charities role owners can enable charity addresses. • The charities' contract is access to anything related to adding and removing charities, in addition to getting what the current charity gets.	**YES, This is valid.**
File 3 PrizeBond.sol • Name: PrizeBond • Symbol: PB • Owner can set the base URI. • A token's owner can also burn his prize bonds at any time and recover the initial deposit.	**YES, This is valid.**
File 4 RandomRequester.sol • This contract is used to generate a random key hash.	**YES, This is valid.**
File 5 Roles.sol • Role contracts can define the roles of fees, migrants, pausers and administrators.	**YES, This is valid.**

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **“Secured”**. Also, these contracts do contain owner control, which does not make them 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 4 low and some very low level issues.
All the issues have been fixed / acknowledged in the revised code.

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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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	Passed
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Passed
	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	Passed
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	Passed
	High consumption ‘for/while’ loop	Moderated
	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**

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Code Quality

This audit scope has 5 smart contract files. Smart contracts contain Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in MyChance Protocol 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 MyChance Protocol.

The MyChance team has provided unit test scripts, which would have helped to determine the integrity of the code in an automated way.

All code parts are not well commented on smart contracts.

Documentation

We were given a MyChance smart contract code in the form of a file. The hash of that code is mentioned above in the table.

As mentioned above, code parts are not well commented. But the logic is straightforward. So it is 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.

Another source of information was its website: https://my-chance.io/ which provided rich information about the project architecture.

Use of Dependencies

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

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

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

AS-IS overview

MyChance.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	checkUpkeep	external	Return value missing	Refer Audit Findings
3	performUpkeep	external	Passed	No Issue
4	pause	write	Return value missing	Refer Audit Findings
5	unpause	write	access only Role	No Issue
6	_updateCallbackGasLimit	write	access only Role	No Issue
7	_approveLP	write	access only Role	No Issue
8	getTotalPrizeBonds	read	Passed	No Issue
9	getStakedAmount	read	Passed	No Issue
10	getListOfTickets	read	Passed	No Issue
11	getTicketData	read	Passed	No Issue
12	getState	read	Passed	No Issue
13	startMigration	write	access only Role	No Issue
14	migrateMyself	external	Passed	No Issue
15	draw	write	Passed	No Issue
16	drawSpecialLottery	external	Passed	No Issue
17	claim	external	Passed	No Issue
18	mintPrizeBond	external	Passed	No Issue
19	burnPrizeBond	external	Passed	No Issue
20	increaseStake	external	Passed	No Issue
21	reduceStake	external	Passed	No Issue
22	canDraw	internal	Passed	No Issue
23	howMuchToClaim	read	Passed	No Issue
24	accumulatedDAI	read	Passed	No Issue
25	accumulatedUSDT	read	Passed	No Issue
26	accumulatedUSDC	read	Passed	No Issue
27	getNextDrawDate	read	Passed	No Issue
28	_executeDraw	internal	Passed	No Issue
29	_executeSpecialDraw	internal	Passed	No Issue
30	fulfillRandomWords	internal	Passed	No Issue
31	setClaimNotRequired	write	access only Role	No Issue
32	setWaitNotRequired	write	access only Role	No Issue
33	_claimFees	write	Passed	No Issue
34	_addSpecialLottery	write	access only Role	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Charities.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	_enableCharity	write	access only Role	No Issue
3	onlyRole	modifier	Passed	No Issue
4	supportsInterface	read	Passed	No Issue
5	hasRole	read	Passed	No Issue
6	_checkRole	internal	Passed	No Issue
7	_checkRole	internal	Passed	No Issue
8	getRoleAdmin	read	Passed	No Issue
9	grantRole	write	access only Role	No Issue
10	revokeRole	write	access only Role	No Issue
11	renounceRole	write	Passed	No Issue
12	_setupRole	internal	Passed	No Issue
13	_setRoleAdmin	internal	Passed	No Issue
14	_grantRole	internal	Passed	No Issue
15	_revokeRole	internal	Passed	No Issue

Roles.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyRole	modifier	Passed	No Issue
3	supportsInterface	read	Passed	No Issue
4	hasRole	read	Passed	No Issue
5	_checkRole	internal	Passed	No Issue
6	_checkRole	internal	Passed	No Issue
7	getRoleAdmin	read	Passed	No Issue
8	grantRole	write	access only Role	No Issue
9	revokeRole	write	access only Role	No Issue
10	renounceRole	write	Passed	No Issue
11	_setupRole	internal	Passed	No Issue
12	_setRoleAdmin	internal	Passed	No Issue
13	_grantRole	internal	Passed	No Issue
14	_revokeRole	internal	Passed	No Issue

PrizeBond.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	onlyMyChance	modifier	Passed	No Issue
3	_baseURI	internal	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

4	_setBaseURI	write	access only Owner	No Issue
5	getAssetType	read	Passed	No Issue
6	setMyChance	external	Passed	No Issue
7	safeMint	external	access only My Chance	No Issue
8	safeBurn	external	access only My Chance	No Issue
9	_beforeTokenTransfer	internal	Passed	No Issue
10	supportsInterface	read	Passed	No Issue
11	supportsInterface	read	Passed	No Issue
12	balanceOf	read	Passed	No Issue
13	ownerOf	read	Passed	No Issue
14	name	read	Passed	No Issue
15	symbol	read	Passed	No Issue
16	tokenURI	read	Passed	No Issue
17	_baseURI	internal	Passed	No Issue
18	approve	write	Passed	No Issue
19	getApproved	read	Passed	No Issue
20	setApprovalForAll	write	Passed	No Issue
21	isApprovedForAll	read	Passed	No Issue
22	transferFrom	write	Passed	No Issue
23	safeTransferFrom	write	Passed	No Issue
24	safeTransferFrom	write	Passed	No Issue
25	_safeTransfer	internal	Passed	No Issue
26	_ownerOf	internal	Passed	No Issue
27	_exists	internal	Passed	No Issue
28	_isApprovedOrOwner	internal	Passed	No Issue
29	_safeMint	internal	Passed	No Issue
30	_safeMint	internal	Passed	No Issue
31	_mint	internal	Passed	No Issue
32	_burn	internal	Passed	No Issue
33	_transfer	internal	Passed	No Issue
34	_approve	internal	Passed	No Issue
35	_setApprovalForAll	internal	Passed	No Issue
36	_requireMinted	internal	Passed	No Issue
37	_checkOnERC721Received	write	Passed	No Issue
38	_beforeTokenTransfer	internal	Passed	No Issue
39	_afterTokenTransfer	internal	Passed	No Issue
40	_beforeConsecutiveTokenTransfer	internal	Passed	No Issue
41	_afterConsecutiveTokenTransfer	internal	Passed	No Issue
42	supportsInterface	read	Passed	No Issue
43	tokenOfOwnerByIndex	read	Passed	No Issue
44	totalSupply	read	Passed	No Issue
45	tokenByIndex	read	Passed	No Issue
46	_beforeTokenTransfer	internal	Passed	No Issue
47	_addTokenToOwnerEnumeration	write	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

48	_addTokenToAllTokensEnumeration	write	Passed	No Issue
49	_removeTokenFromOwnerEnumeration	write	Passed	No Issue
50	_removeTokenFromAllTokensEnumeration	write	Passed	No Issue
51	owner	read	Passed	No Issue
52	onlyOwner	modifier	Passed	No Issue
53	renounceOwnership	write	access only Owner	No Issue
54	transferOwnership	write	access only Owner	No Issue

RandomRequester.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	_randomnessRequest	internal	Passed	No Issue
3	fulfillRandomWords	internal	Passed	No Issue
4	rawFulfillRandomWords	external	Passed	No Issue

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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 loss
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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
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

(1) Function input parameters lack of check: **PrizeBond.sol**
Variable validation is not performed in below functions:

- setMyChance = _myChance, _myChanceMigration.

Resolution: We advise to put validation: integer type variables should be greater than 0 and address type variables should not be address(0).
Status: Fixed

(2) Infinite Loop: **Charities.sol**
In _enableCharity function, the for loop does not have aCharities upper length limit, which costs more gas.

Resolution: Upper bound should have a certain limit in for loops.
Status: Acknowledged

(3) Critical operation lacks event log: **MyChance.sol**
Missing event log for:

- draw
- drawSpecialLottery
- burnPrizeBond

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

- increaseStake
- reduceStake
- _claimFees

Resolution: Write an event log for listed events.
Status: Fixed

(4) Return value missing: **MyChance.sol**



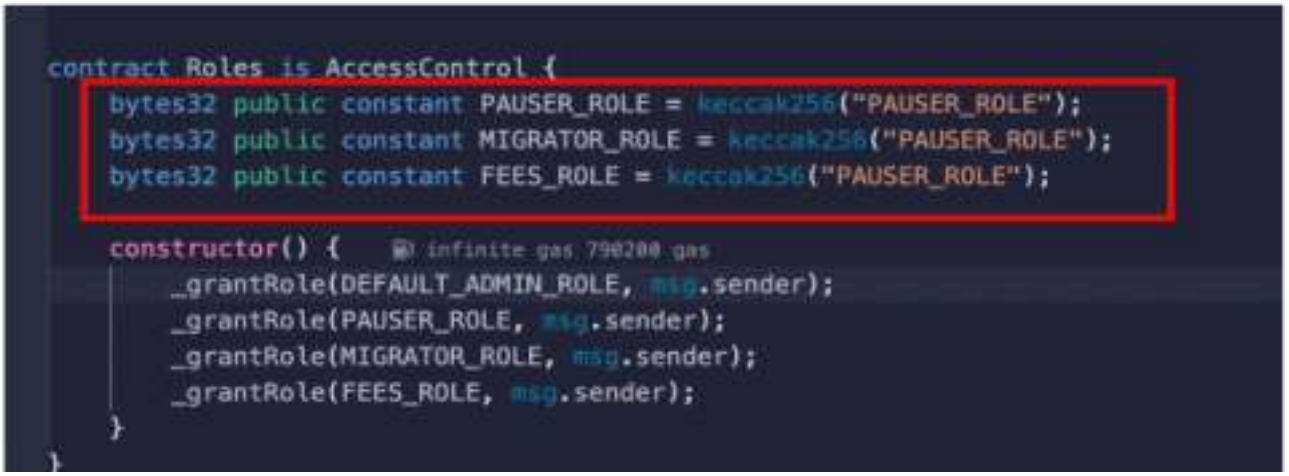
In the checkUpkeep function, Unnamed return variable can remain unassigned. Add an explicit return with value to all non-reverting code paths or name the variable.

Resolution: We suggest adding a "return" value in the function.
Status: Acknowledged

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Very Low / Informational / Best practices:

(1) Same string is passed for all 3 constants: **Roles.sol**



For all 3 PAUSER, MIGRATOR,FEES roles the same hash is generated by passing the same ""PAUSER_ROLE"" string to keccak256 method. Which consumes more gas. And Transaction Fee.

Gas and TX fee consumption comparison:

- The contract with 3 keccak256 - 960,231 gas. Transaction Fee: 0.0264063525 AVAX ($0.60).
- The contract with 1 keccak256 - 924,554 gas, Transaction Fee: 0.025425235 AVAX ($0.58).

Resolution: Same hash is generated for all the roles, instead generate once and use for all or have different hash for different roles.

Status: Fixed

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Centralization

This smart contract has 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:

- _enableCharity: Charities role owner can enable charity address status.
- pause: MyChance role owner can trigger a stopped state.
- unpause: MyChance role owner can return to normal state.
- _updateCallbackGasLimit: MyChance role owner can update callback gas limit values.
- _approveLP: MyChance role owner can approve LP tokens.
- _setBaseURI: PrizeBond owner can set a base URI.
- setMyChance: PrizeBond owner can set my chance address and my chance migration.
- safeMint: PrizeBond MyChance owner can safely mint tokens.
- safeBurn: PrizeBond MyChance owner can safely burn tokens.

To make the smart contract 100% decentralized, we suggest renouncing ownership in the smart contract once its function is completed.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Conclusion

We were given a contract code in the form of a file. And we have used all possible tests based on given objects as files. We have observed 4 low severity issues, and some Informational issues in smart contracts. All the issues have been fixed / acknowledged in the revised code. **So, the smart contracts are ready for the mainnet deployment.**

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 **"Secure".**

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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

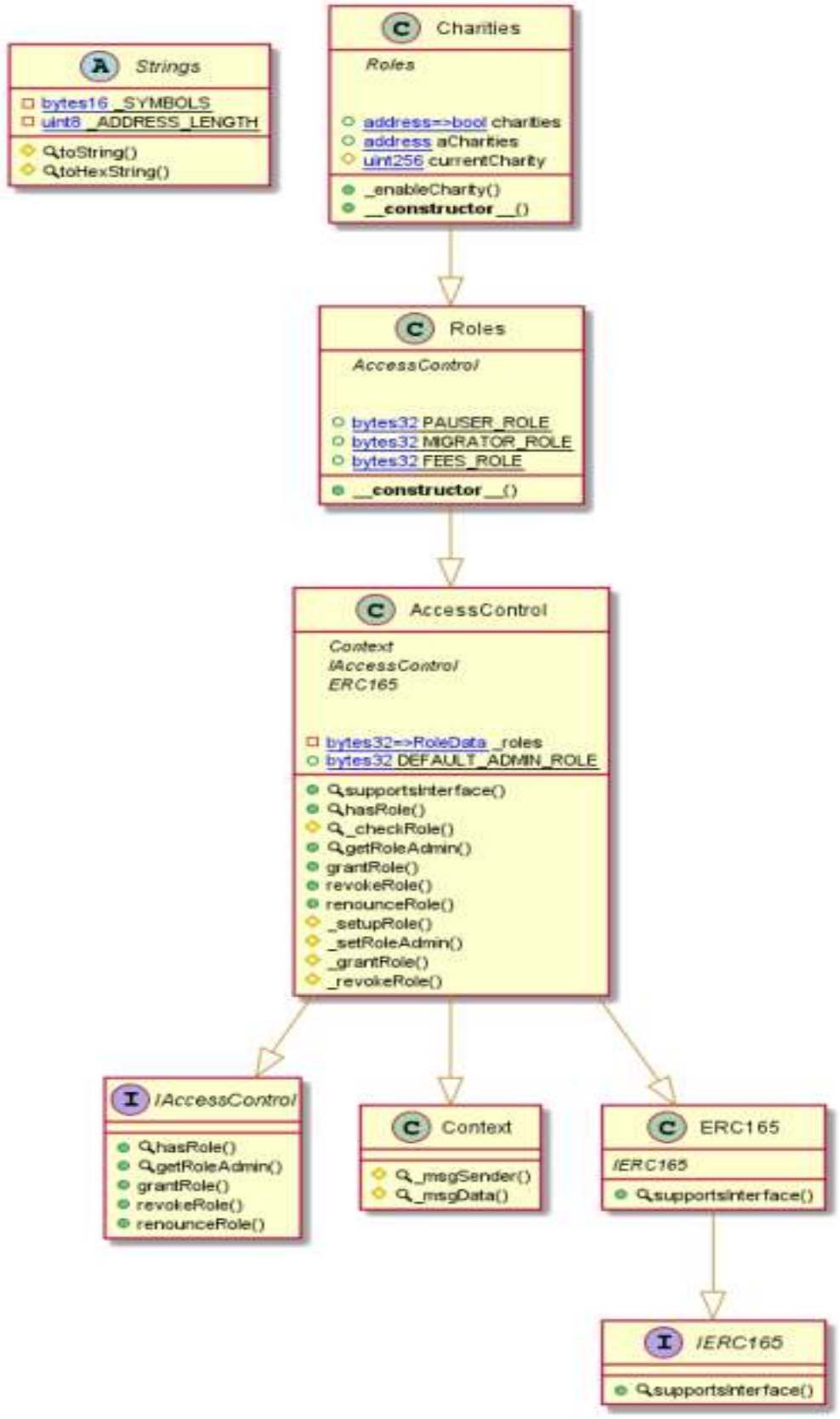
Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Appendix

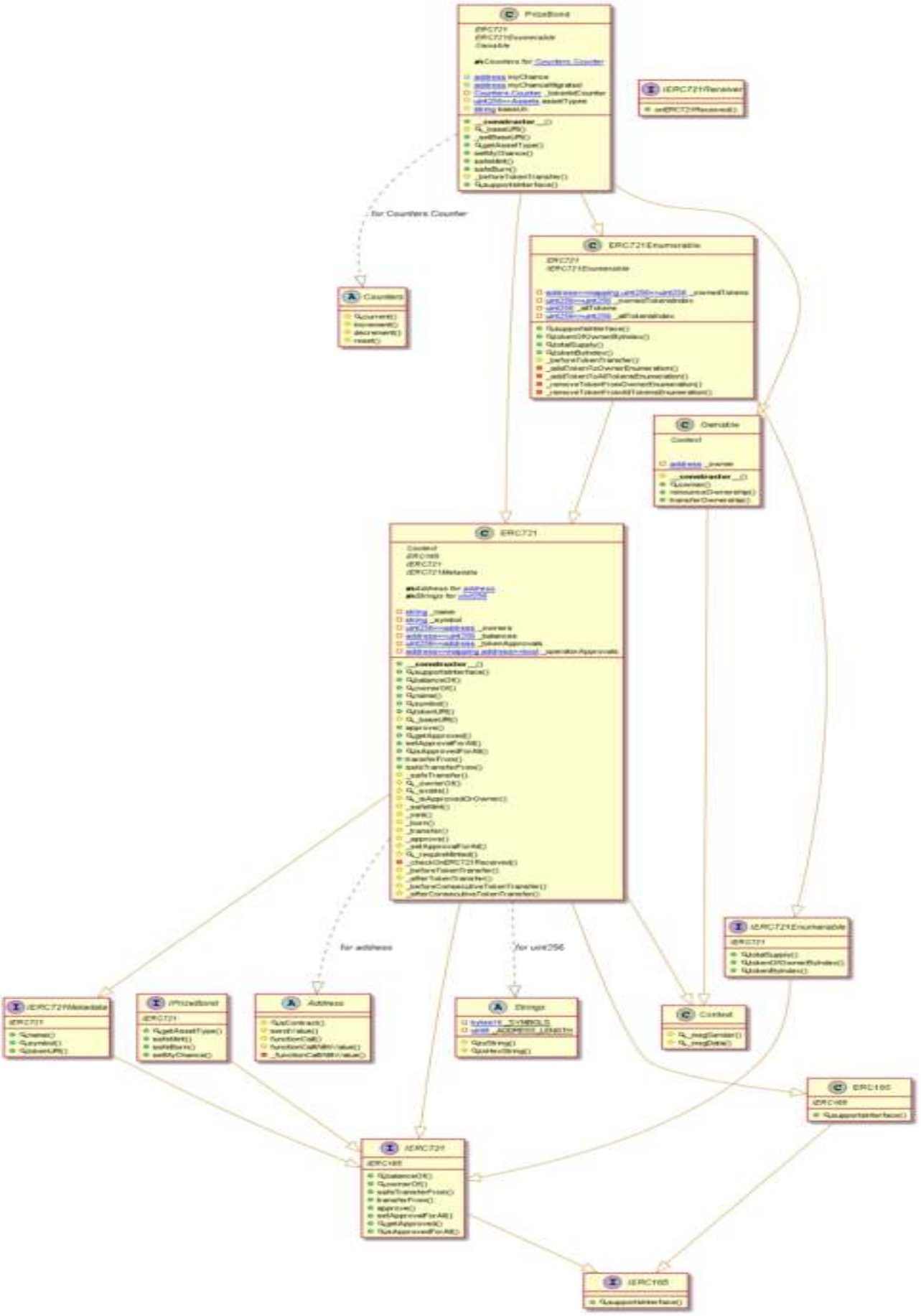
Code Flow Diagram - MyChance Protocol

Charities Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

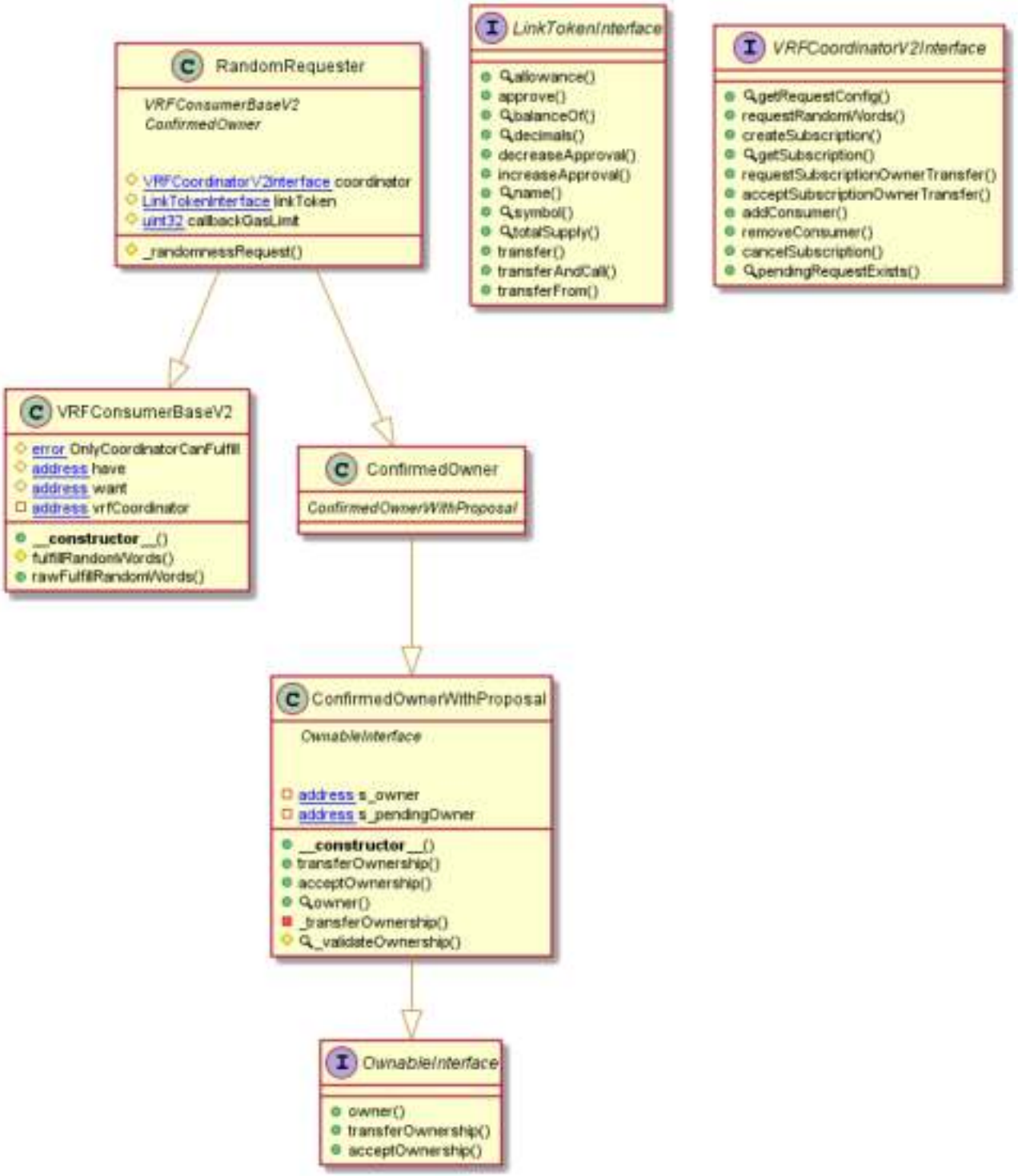
PrizeBond Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

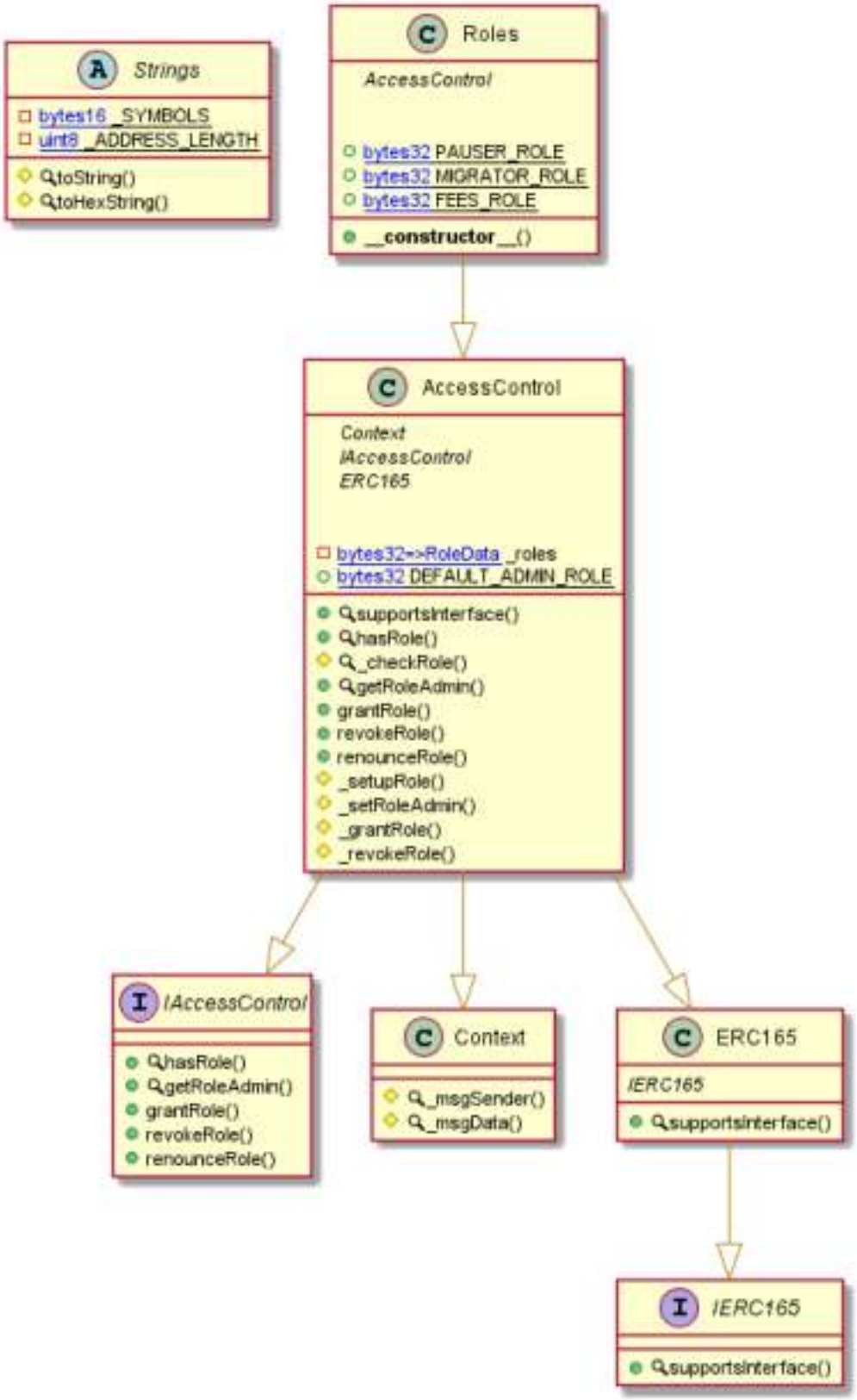
Email: audit@EtherAuthority.io

RandomRequester Diagram



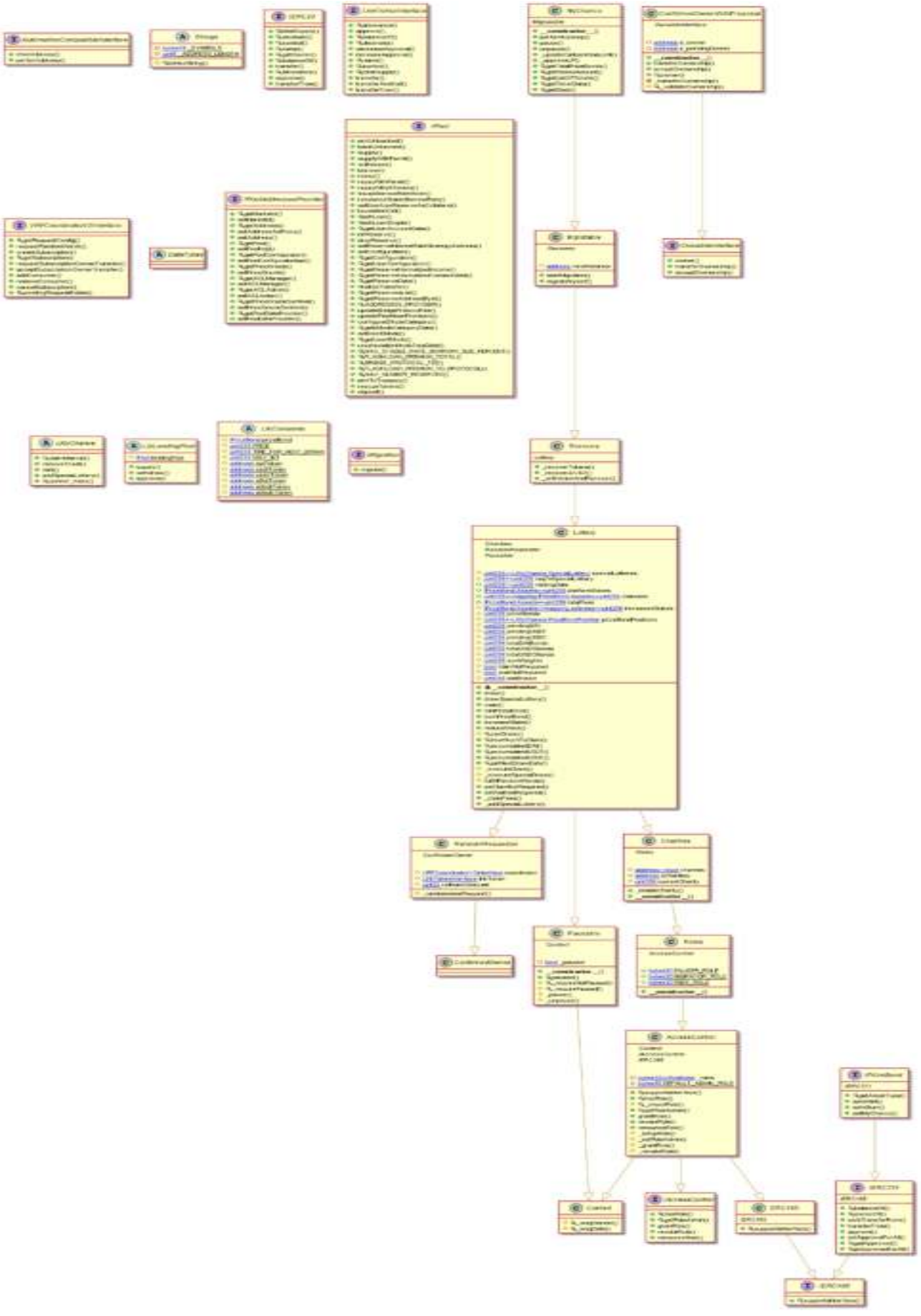
This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Roles Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

MyChance Diagram



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Slither Results Log

Slither log >> MyChance.sol

```
INFO:Detectors:
Migratable.migrateMyself(uint256) (MyChance.sol#1553-1588) should emit an event for:
        - sumWeights -= deletedWeght (MyChance.sol#1587)
Lottery.burnPrizeBond(uint256) (MyChance.sol#1331-1358) should emit an event for:
        - sumWeights -= deletedWeght (MyChance.sol#1357)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
Migratable.startMigration(address)._newInstance (MyChance.sol#1548) lacks a zero-check on :
                - newInstance = _newInstance (MyChance.sol#1550)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in Lottery._claimFees() (MyChance.sol#1478-1502):
        External calls:
        - LibLendingPool.withdraw(LibConstants.daiToken,daiFees,msg.sender) (MyChance.sol#1489)
        State variables written after the call(s):
        - pendingDAI -= daiFees (MyChance.sol#1490)
Reentrancy in Lottery._claimFees() (MyChance.sol#1478-1502):
        External calls:
        - LibLendingPool.withdraw(LibConstants.daiToken,daiFees,msg.sender) (MyChance.sol#1489)
        - LibLendingPool.withdraw(LibConstants.usdtToken,usdtFees,msg.sender) (MyChance.sol#1494)
        State variables written after the call(s):
        - pendingUSDT -= usdtFees (MyChance.sol#1495)
Reentrancy in Lottery._claimFees() (MyChance.sol#1478-1502):
        External calls:
        - LibLendingPool.withdraw(LibConstants.daiToken,daiFees,msg.sender) (MyChance.sol#1489)
        - LibLendingPool.withdraw(LibConstants.usdtToken,usdtFees,msg.sender) (MyChance.sol#1494)
        - LibLendingPool.withdraw(LibConstants.usdcToken,usdcFees,msg.sender) (MyChance.sol#1499)
        State variables written after the call(s):
        - pendingUSDC -= usdcFees (MyChance.sol#1500)
Reentrancy in Lottery.burnPrizeBond(uint256) (MyChance.sol#1331-1358):
        External calls:
        - LibConstants.prizeBond.safeBurn(_tokenId) (MyChance.sol#1337)
        - LibLendingPool.withdraw(LibConstants.daiToken,LibConstants.PRICE * weight * 1e18,msg.sender) (MyChance.sol#1342)
        State variables written after the call(s):
        - totalDAIBonds -= weight (MyChance.sol#1343)
Reentrancy in Lottery.burnPrizeBond(uint256) (MyChance.sol#1331-1358):
        External calls:
```

```
        - LibConstants.prizeBond.safeBurn(_tokenId) (MyChance.sol#1337)
        - LibLendingPool.withdraw(LibConstants.usdcToken,LibConstants.PRICE * weight * 1e6,msg.sender) (MyChance.sol#1346)
        State variables written after the call(s):
        - totalUSDCBonds -= weight (MyChance.sol#1347)
Reentrancy in Lottery.burnPrizeBond(uint256) (MyChance.sol#1331-1358):
        External calls:
        - LibConstants.prizeBond.safeBurn(_tokenId) (MyChance.sol#1337)
        - LibLendingPool.withdraw(LibConstants.usdtToken,LibConstants.PRICE * weight * 1e6,msg.sender) (MyChance.sol#1350)
        State variables written after the call(s):
        - totalUSDTBonds -= weight (MyChance.sol#1351)
Reentrancy in Lottery.burnPrizeBond(uint256) (MyChance.sol#1331-1358):
        External calls:
        - LibConstants.prizeBond.safeBurn(_tokenId) (MyChance.sol#1337)
        - LibLendingPool.withdraw(LibConstants.daiToken,LibConstants.PRICE * weight * 1e18,msg.sender) (MyChance.sol#1342)
        - LibLendingPool.withdraw(LibConstants.usdcToken,LibConstants.PRICE * weight * 1e6,msg.sender) (MyChance.sol#1346)
        - LibLendingPool.withdraw(LibConstants.usdtToken,LibConstants.PRICE * weight * 1e6,msg.sender) (MyChance.sol#1350)
        State variables written after the call(s):
        - sumWeights -= deletedWeght (MyChance.sol#1357)
Reentrancy in Lottery.claim(uint256,uint256) (MyChance.sol#1237-1301):
        External calls:
        - LibLendingPool.withdraw(LibConstants.daiToken,claimVariables.withdrawalAmountDAI,address(this)) (MyChance.sol#1266)
        - require(bool,string)(IERC20(LibConstants.daiToken).transfer(msg.sender,(claimVariables.withdrawalAmountDAI - claimVa
riables.totalToCharityDAI)),Transfer failed) (MyChance.sol#1267)
        - require(bool,string)(IERC20(LibConstants.daiToken).transfer(charity,claimVariables.totalToCharityDAI),Transfer faile
d) (MyChance.sol#1268)
        State variables written after the call(s):
        - pendingUSDT -= claimVariables.withdrawalAmountUSDT (MyChance.sol#1274)
Reentrancy in Lottery.claim(uint256,uint256) (MyChance.sol#1237-1301):
        External calls:
        - LibLendingPool.withdraw(LibConstants.daiToken,claimVariables.withdrawalAmountDAI,address(this)) (MyChance.sol#1266)
        - require(bool,string)(IERC20(LibConstants.daiToken).transfer(msg.sender,(claimVariables.withdrawalAmountDAI - claimVa
riables.totalToCharityDAI)),Transfer failed) (MyChance.sol#1267)
        - require(bool,string)(IERC20(LibConstants.daiToken).transfer(charity,claimVariables.totalToCharityDAI),Transfer faile
d) (MyChance.sol#1268)
        - LibLendingPool.withdraw(LibConstants.usdtToken,claimVariables.withdrawalAmountUSDT,address(this)) (MyChance.sol#1275
)
```

```
        - require(bool,string)(IERC20(LibConstants.usdtToken).transfer(msg.sender,(claimVariables.withdrawalAmountUSDT - claim
Variables.totalToCharityUSDT)),Transfer failed) (MyChance.sol#1276)
        - require(bool,string)(IERC20(LibConstants.usdtToken).transfer(charity,claimVariables.totalToCharityUSDT),Transfer fai
led) (MyChance.sol#1277)
        State variables written after the call(s):
        - pendingUSDC -= claimVariables.withdrawalAmountUSDC (MyChance.sol#1283)
Reentrancy in Lottery.increaseStake(IPrizeBond.Assets,uint256) (MyChance.sol#1360-1378):
        External calls:
        - require(bool,string)(IERC20(LibConstants.daiToken).transferFrom(msg.sender,address(this),_total),Transfer failed) (M
yChance.sol#1362)
        - LibLendingPool.supply(LibConstants.daiToken,_total) (MyChance.sol#1363)
        - require(bool,string)(IERC20(LibConstants.usdcToken).transferFrom(msg.sender,address(this),_total),Transfer failed) (
MyChance.sol#1366)
        - LibLendingPool.supply(LibConstants.usdcToken,_total) (MyChance.sol#1367)
        - require(bool,string)(IERC20(LibConstants.usdtToken).transferFrom(msg.sender,address(this),_total),Transfer failed) (
MyChance.sol#1370)
        - LibLendingPool.supply(LibConstants.usdtToken,_total) (MyChance.sol#1371)
        State variables written after the call(s):
        - increasedStakes[_assetType][msg.sender] += _total (MyChance.sol#1377)
        - platformStakes[_assetType] += _total (MyChance.sol#1376)
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

```
        - LibLendingPool.supply(LibConstants.daiToken,cost) (MyChance.sol#1309)
        - require(bool,string)(IERC20(LibConstants.usdcToken).transferFrom(msg.sender,address(this),cost_scope_0),Transfer fai
led) (MyChance.sol#1314)
        - LibLendingPool.supply(LibConstants.usdcToken,cost_scope_0) (MyChance.sol#1315)
        - require(bool,string)(IERC20(LibConstants.usdtToken).transferFrom(msg.sender,address(this),cost_scope_1),Transfer fai
led) (MyChance.sol#1320)
        - LibLendingPool.supply(LibConstants.usdtToken,cost_scope_1) (MyChance.sol#1321)
        - LMyChance.mint(LibConstants.prizeBond,_assetType,weight,mintingDate,prizeBonds,prizeBondPositions) (MyChance.sol#132
7)
        State variables written after the call(s):
        - sumWeights += weight (MyChance.sol#1328)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in Lottery._addSpecialLottery(uint256,IPrizeBond.Assets,uint256,string) (MyChance.sol#1504-1524):
        External calls:
        - require(bool,string)(IERC20(token).transferFrom(msg.sender,address(this),_total),Transfer failed) (MyChance.sol#1520
)
        - LibLendingPool.supply(token,_total) (MyChance.sol#1522)
        Event emitted after the call(s):
        - NewSpecialDraw(_drawDate) (MyChance.sol#1523)
Reentrancy in Lottery.claim(uint256,uint256) (MyChance.sol#1237-1301):
        External calls:
        - LibLendingPool.withdraw(LibConstants.daiToken,claimVariables.withdrawalAmountDAI,address(this)) (MyChance.sol#1266)
        - require(bool,string)(IERC20(LibConstants.daiToken).transfer(msg.sender,(claimVariables.withdrawalAmountDAI - claimVa
riables.totalToCharityDAI)),Transfer failed) (MyChance.sol#1267)
        - require(bool,string)(IERC20(LibConstants.daiToken).transfer(charity,claimVariables.totalToCharityDAI),Transfer faile
d) (MyChance.sol#1268)
        - LibLendingPool.withdraw(LibConstants.usdtToken,claimVariables.withdrawalAmountUSDT,address(this)) (MyChance.sol#1275
)
        - require(bool,string)(IERC20(LibConstants.usdtToken).transfer(msg.sender,(claimVariables.withdrawalAmountUSDT - claim
Variables.totalToCharityUSDT)),Transfer failed) (MyChance.sol#1276)
        - require(bool,string)(IERC20(LibConstants.usdtToken).transfer(charity,claimVariables.totalToCharityUSDT),Transfer fai
led) (MyChance.sol#1277)
        - LibLendingPool.withdraw(LibConstants.usdcToken,claimVariables.withdrawalAmountUSDC,address(this)) (MyChance.sol#1284
)
        - require(bool,string)(IERC20(LibConstants.usdcToken).transfer(msg.sender,(claimVariables.withdrawalAmountUSDC - claim
Variables.totalToCharityUSDC)),Transfer failed) (MyChance.sol#1285)
```

```
INFO:Detectors:
Variable Lottery.mintPrizeBond(IPrizeBond.Assets,uint256).cost_scope_0 (MyChance.sol#1313) is too similar to Lottery.mintPrize
Bond(IPrizeBond.Assets,uint256).cost_scope_1 (MyChance.sol#1319)
Variable Lottery.pendingUSDC (MyChance.sol#1183) is too similar to Lottery.pendingUSDT (MyChance.sol#1182)
Variable Lottery.totalUSDCBonds (MyChance.sol#1187) is too similar to Lottery.totalUSDTBonds (MyChance.sol#1186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
MyChance.slitherConstructorVariables() (MyChance.sol#1591-1639) uses literals with too many digits:
        - callbackGasLimit = 800000 (MyChance.sol#325)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
RandomRequester.coordinator (MyChance.sol#323) is never used in MyChance (MyChance.sol#1591-1639)
RandomRequester.linkToken (MyChance.sol#324) is never used in MyChance (MyChance.sol#1591-1639)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (MyChance.sol#114-116)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (MyChance.sol#118-120)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (MyChance.sol#122-126)
transferOwnership(address) should be declared external:
        - ConfirmedOwnerWithProposal.transferOwnership(address) (MyChance.sol#284-286)
owner() should be declared external:
        - ConfirmedOwnerWithProposal.owner() (MyChance.sol#298-300)
setClaimNotRequired(bool) should be declared external:
        - Lottery.setClaimNotRequired(bool) (MyChance.sol#1470-1472)
setWaitNotRequired(bool) should be declared external:
        - Lottery.setWaitNotRequired(bool) (MyChance.sol#1474-1476)
_claimFees() should be declared external:
        - Lottery._claimFees() (MyChance.sol#1478-1502)
_addSpecialLottery(uint256,IPrizeBond.Assets,uint256,string) should be declared external:
        - Lottery._addSpecialLottery(uint256,IPrizeBond.Assets,uint256,string) (MyChance.sol#1504-1524)
_recoverTokens(uint256,address) should be declared external:
        - Recovery._recoverTokens(uint256,address) (MyChance.sol#1530-1533)
_recoverAVAX(uint256) should be declared external:
        - Recovery._recoverAVAX(uint256) (MyChance.sol#1535-1537)
```

```
_withdrawAndRecover(uint256,address) should be declared external:
        - Recovery._withdrawAndRecover(uint256,address) (MyChance.sol#1539-1542)
startMigration(address) should be declared external:
        - Migratable.startMigration(address) (MyChance.sol#1548-1551)
pause() should be declared external:
        - MyChance.pause() (MyChance.sol#1603-1605)
unpause() should be declared external:
        - MyChance.unpause() (MyChance.sol#1607-1609)
_updateCallbackGasLimit(uint32) should be declared external:
        - MyChance._updateCallbackGasLimit(uint32) (MyChance.sol#1611-1613)
getTotalPrizeBonds() should be declared external:
        - MyChance.getTotalPrizeBonds() (MyChance.sol#1619-1621)
getStakedAmount(IPrizeBond.Assets) should be declared external:
        - MyChance.getStakedAmount(IPrizeBond.Assets) (MyChance.sol#1623-1625)
getListOfTickets() should be declared external:
        - MyChance.getListOfTickets() (MyChance.sol#1627-1629)
getTicketData(uint256) should be declared external:
        - MyChance.getTicketData(uint256) (MyChance.sol#1631-1633)
getState() should be declared external:
        - MyChance.getState() (MyChance.sol#1635-1637)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:MyChance.sol analyzed (30 contracts with 75 detectors), 134 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Slither log >> Charities.sol

```
INFO:Detectors:
Strings.toString(uint256) (Charities.sol#9-23) uses assembly
        - INLINE ASM (Charities.sol#12-13)
        - INLINE ASM (Charities.sol#16-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
AccessControl._setRoleAdmin(bytes32,bytes32) (Charities.sol#147-151) is never used and should be removed
AccessControl._setupRole(bytes32,address) (Charities.sol#143-145) is never used and should be removed
Context._msgData() (Charities.sol#73-75) is never used and should be removed
Strings.toHexString(uint256) (Charities.sol#25-28) is never used and should be removed
Strings.toString(uint256) (Charities.sol#9-23) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.4 (Charities.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function Charities._enableCharity(address,bool) (Charities.sol#188-207) is not in mixedCase
Parameter Charities._enableCharity(address,bool)._charity (Charities.sol#188) is not in mixedCase
Parameter Charities._enableCharity(address,bool)._enabled (Charities.sol#188) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
INFO:Detectors:
Charities.currentCharity (Charities.sol#186) is never used in Charities (Charities.sol#183-217)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
Charities.currentCharity (Charities.sol#186) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (Charities.sol#129-131)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (Charities.sol#133-135)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (Charities.sol#137-141)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Charities.sol analyzed (8 contracts with 75 detectors), 16 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> PrizeBond.sol

```
INFO:Detectors:
Pragma version^0.8.4 (PrizeBond.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (PrizeBond.sol#43-48):
        - (success) = recipient.call{value: amount}() (PrizeBond.sol#46)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (PrizeBond.sol#80-102):
        - (success,returndata) = target.call{value: weiValue}(data) (PrizeBond.sol#88)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function PrizeBond._setBaseURI(string) (PrizeBond.sol#671-673) is not in mixedCase
Parameter PrizeBond._setBaseURI(string)._baseUri (PrizeBond.sol#671) is not in mixedCase
Parameter PrizeBond.setMyChance(address,address)._myChance (PrizeBond.sol#679) is not in mixedCase
Parameter PrizeBond.setMyChance(address,address)._myChanceMigration (PrizeBond.sol#679) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (PrizeBond.sol#200)" inContext (PrizeBond.sol#195-203)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (PrizeBond.sol#225-228)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (PrizeBond.sol#230-234)
name() should be declared external:
        - ERC721.name() (PrizeBond.sol#282-284)
symbol() should be declared external:
        - ERC721.symbol() (PrizeBond.sol#286-288)
tokenURI(uint256) should be declared external:
        - ERC721.tokenURI(uint256) (PrizeBond.sol#290-295)
approve(address,uint256) should be declared external:
        - ERC721.approve(address,uint256) (PrizeBond.sol#301-311)
setApprovalForAll(address,bool) should be declared external:
        - ERC721.setApprovalForAll(address,bool) (PrizeBond.sol#319-321)
```

```
transferFrom(address,address,uint256) should be declared external:
        - ERC721.transferFrom(address,address,uint256) (PrizeBond.sol#327-335)
safeTransferFrom(address,address,uint256) should be declared external:
        - ERC721.safeTransferFrom(address,address,uint256) (PrizeBond.sol#337-343)
tokenOfOwnerByIndex(address,uint256) should be declared external:
        - ERC721Enumerable.tokenOfOwnerByIndex(address,uint256) (PrizeBond.sol#552-555)
tokenByIndex(uint256) should be declared external:
        - ERC721Enumerable.tokenByIndex(uint256) (PrizeBond.sol#561-564)
_setBaseURI(string) should be declared external:
        - PrizeBond._setBaseURI(string) (PrizeBond.sol#671-673)
getAssetType(uint256) should be declared external:
        - PrizeBond.getAssetType(uint256) (PrizeBond.sol#675-677)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:PrizeBond.sol analyzed (15 contracts with 75 detectors), 47 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Slither log >> Roles.sol

```
INFO:Detectors:
Strings.toString(uint256) (Roles.sol#8-22) uses assembly
        - INLINE ASM (Roles.sol#11-12)
        - INLINE ASM (Roles.sol#15-17)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
AccessControl._setRoleAdmin(bytes32,bytes32) (Roles.sol#146-150) is never used and should be removed
AccessControl._setupRole(bytes32,address) (Roles.sol#142-144) is never used and should be removed
Context._msgData() (Roles.sol#72-74) is never used and should be removed
Strings.toHexString(uint256) (Roles.sol#24-27) is never used and should be removed
Strings.toString(uint256) (Roles.sol#8-22) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.4 (Roles.sol#1) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (Roles.sol#128-130)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (Roles.sol#132-134)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (Roles.sol#136-140)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Roles.sol analyzed (7 contracts with 75 detectors), 11 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> RandomRequester.sol

```
INFO:Detectors:
RandomRequester._randomnessRequest() (RandomRequester.sol#149-151) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.4 (RandomRequester.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7
.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Variable ConfirmedOwnerWithProposal.s_owner (RandomRequester.sol#91) is not in mixedCase
Variable ConfirmedOwnerWithProposal.s_pendingOwner (RandomRequester.sol#92) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
RandomRequester.slitherConstructorVariables() (RandomRequester.sol#142-153) uses literals with too many digits:
        - callbackGasLimit = 800000 (RandomRequester.sol#145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
RandomRequester.coordinator (RandomRequester.sol#143) is never used in RandomRequester (RandomRequester.sol#142-153)
RandomRequester.linkToken (RandomRequester.sol#144) is never used in RandomRequester (RandomRequester.sol#142-153)
RandomRequester.callbackGasLimit (RandomRequester.sol#145) is never used in RandomRequester (RandomRequester.sol#142-153)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
INFO:Detectors:
INFO:Slither:RandomRequester.sol analyzed (5 contracts with 75 detectors), 22 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Solidity Static Analysis

MyChance.sol

Security

Transaction origin:

Use of tx.origin: "tx.origin" is useful only in very exceptional cases. If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.

more

Pos: 12:8:

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: 103:17:

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: 265:15:

Gas & Economy

Gas costs:

Gas requirement of function MyChance.grantRole 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: 144:4:

Gas costs:

Gas requirement of function Roles.renounceRole 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: 179:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

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: 23:12:

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 11:2:

Miscellaneous

Similar variable names:

MyChance.getState() : Variables have very similar names "totalUSDTBonds" and "totalUSDCBonds". Note: Modifiers are currently not considered by this static analysis.
Pos: 512:69:

Similar variable names:

Migratable.migrateMyself(uint256) : Variables have very similar names "totalUSDTBonds" and "totalUSDCBonds". Note: Modifiers are currently not considered by this static analysis.
Pos: 439:12:

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: 451:8:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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: 41:8:

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: 27:33:

Charities.sol

Gas & Economy

Gas costs:

Gas requirement of function Charities.getRoleAdmin 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: 128:4:

Gas costs:

Gas requirement of function Roles.grantRole 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: 144:4:

Gas costs:

Gas requirement of function Charities.revokeRole 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: 159:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Gas costs:

Gas requirement of function Charities._enableCharity 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: 14:4:

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: 23:12:

Miscellaneous

Constant/View/Pure functions:

Strings.toHexString(address) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 72:4:

Constant/View/Pure functions:

ERC165.supportsInterface(bytes4) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 26:4:

Similar variable names:

Charities._enableCharity(address,bool) : Variables have very similar names "charities" and "aCharities". Note: Modifiers are currently not considered by this static analysis.
Pos: 31:12:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

No return:

IERC165.supportsInterface(bytes4): Defines a return type but never explicitly returns a value.
Pos: 24:4:

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: 65:8:

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: 15:8:

PrizeBond.sol

Security

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 128:4:

Inline assembly:

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.
more
Pos: 213:16:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

Low level calls:

Use of "delegatecall": should be avoided whenever possible. External code, that is called can change the state of the calling contract and send ether from the caller's balance. If this is wanted behaviour, use the Solidity library feature if possible.

more

Pos: 191:50:

Gas & Economy

Gas costs:

Gas requirement of function ERC721.name 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: 79:4:

Gas costs:

Gas requirement of function PrizeBond.setMyChance 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: 45:4:

Gas costs:

Gas requirement of function PrizeBond.safeBurn 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: 58:4:

Miscellaneous

Constant/View/Pure functions:

ERC721.supportsInterface(bytes4) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.

more

Pos: 52:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

Constant/View/Pure functions:

PrizeBond._beforeTokenTransfer(address,address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 64:4:

Similar variable names:

ERC721Enumerable.tokenOfOwnerByIndex(address,uint256) : Variables have very similar names "_owners" and "owner". Note: Modifiers are currently not considered by this static analysis.
Pos: 39:28:

No return:

IPrizeBond.safeMint(address,enum IPrizeBond.Assets): Defines a return type but never explicitly returns a value.
Pos: 13:4:

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: 136:8:

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: 25:8:

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: 160:8:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

RandomRequester.sol

ERC

ERC20:

ERC20 contract's "decimals" function should have "uint8" as return type
more
Pos: 11:2:

Miscellaneous

Constant/View/Pure functions:

VRFConsumerBaseV2.fulfillRandomWords(uint256,uint256[]) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 122:2:

Constant/View/Pure functions:

VRFCoordinatorV2Interface.cancelSubscription(uint64,address) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 115:2:

Similar variable names:

ConfirmedOwnerWithProposal.(address,address) : Variables have very similar names "s_pendingOwner" and "pendingOwner". Note: Modifiers are currently not considered by this static analysis.
Pos: 22:25:

No return:

VRFCoordinatorV2Interface.createSubscription(): Defines a return type but never explicitly returns a value.
Pos: 61:2:

No return:

VRFCoordinatorV2Interface.getSubscription(uint64): Defines a return type but never explicitly returns a value.
Pos: 71:2:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

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: 58:4:

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: 69:4:

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io

Solhint Linter

MyChance.sol

```
MyChance.sol:3:1: Error: Compiler version ^0.8.15 does not satisfy
the r semver requirement
MyChance.sol:6:21: Error: Use double quotes for string literals
MyChance.sol:7:26: Error: Use double quotes for string literals
MyChance.sol:26:1: Error: Contract has 19 states declarations but
allowed no more than 15
MyChance.sol:53:5: Error: Explicitly mark visibility of state
MyChance.sol:73:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity >=0.7.0)
MyChance.sol:103:18: Error: Avoid to make time-based decisions in
your business logic
MyChance.sol:130:146: Error: Use double quotes for string literals
MyChance.sol:131:104: Error: Use double quotes for string literals
MyChance.sol:135:13: Error: Possible reentrancy vulnerabilities.
Avoid state changes after transfer.MyChance.sol:137:13: Error:
Possible reentrancy vulnerabilities. Avoid state changes after
transfer.
MyChance.sol:139:149: Error: Use double quotes for string literals
MyChance.sol:140:106: Error: Use double quotes for string literals
MyChance.sol:144:13: Error: Possible reentrancy vulnerabilities.
Avoid state changes after transfer.
MyChance.sol:145:13: Error: Possible reentrancy vulnerabilities.
Avoid state changes after transfer.
MyChance.sol:146:13: Error: Possible reentrancy vulnerabilities.
Avoid state changes after transfer.
MyChance.sol:230:101: Error: Use double quotes for string literals
MyChance.sol:234:101: Error: Use double quotes for string literals
MyChance.sol:265:16: Error: Avoid to make time-based decisions in
your business logic
MyChance.sol:337:74: Error: Visibility modifier must be first in list
of modifiers
MyChance.sol:341:73: Error: Visibility modifier must be first in list
of modifiers
MyChance.sol:345:47: Error: Visibility modifier must be first in list
of modifiers
MyChance.sol:387:80: Error: Use double quotes for string literals
MyChance.sol:393:32: Error: Code contains empty blocks
MyChance.sol:399:63: Error: Use double quotes for string literals
MyChance.sol:413:5: Error: Explicitly mark visibility of state
MyChance.sol:449:61: Error: Use double quotes for string literals
MyChance.sol:460:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
```

Charities.sol

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io

```
Charities.sol:3:1: Error: Compiler version ^0.8.15 does not satisfy
the r semver requirement
Charities.sol:12:5: Error: Explicitly mark visibility of state
Charities.sol:35:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
```

PrizeBond.sol

```
PrizeBond.sol:3:1: Error: Compiler version ^0.8.15 does not satisfy
the r semver requirement
PrizeBond.sol:22:5: Error: Explicitly mark visibility of state
PrizeBond.sol:29:5: Error: Explicitly mark visibility in function
(Set ignoreConstructors to true if using solidity >=0.7.0)
PrizeBond.sol:29:45: Error: Code contains empty blocks
PrizeBond.sol:31:5: Error: Explicitly mark visibility of state
```

RandomRequester.sol

```
RandomRequester.sol:3:1: Error: Compiler version ^0.8.15 does not
satisfy the r semver requirement
RandomRequester.sol:14:5: Error: Explicitly mark visibility of state
RandomRequester.sol:14:22: Error: Constant name must be in
capitalized SNAKE_CASE
RandomRequester.sol:15:5: Error: Explicitly mark visibility of state
RandomRequester.sol:15:22: Error: Constant name must be in
capitalized SNAKE_CASE
RandomRequester.sol:16:5: Error: Explicitly mark visibility of state
RandomRequester.sol:18:6: Error: Explicitly mark visibility in
function (Set ignoreConstructors to true if using solidity >=0.7.0)
```

Roles.sol

```
Roles.sol:3:1: Error: Compiler version ^0.8.15 does not satisfy the r
semver requirement
Roles.sol:14:5: Error: Explicitly mark visibility in function (Set
ignoreConstructors to true if using solidity >=0.7.0)
```

Software analysis result:

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

This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.
Email: audit@EtherAuthority.io



This is a private and confidential document. No part of this document should be disclosed to third party without prior written permission of EtherAuthority.

Email: audit@EtherAuthority.io