Cyberscope

Audit Report

CompBet

October 2022

SHA56 7d82a5cf0bd8a58345421e33575cfaabfc82f39ba826083bf4d1b4e42829b267

Audited by © cyberscope

Table of Contents

Audit Updates

Initial Audit	25rd September 2022 https://github.com/cyberscope-io/audits/blob/main/ABP/v1/audit.pdf
Corrected	14th October 2022

Source Files

Filename	SHA256
@chainlink/contracts/src/v0.8/interfaces/VRFCoordinatorV2Interface.sol	1e120dcde1dc1cd1c5fe98e96e3c2cfb245e02b5d5e685a3216f553c9dbc9ed4
@chainlink/contracts/src/v0.8/VRFConsumerBaseV2.sol	c172fc2a79410c284709406356a6bdebaf504b647022bd1b3435787f987163e7
@openzeppelin/contracts/access/Ownable.sol	9353af89436556f7ba8abb3f37a6677249aa4df6024fbfaa94f79ab2f44f3231
@openzeppelin/contracts/security/ReentrancyGuard.sol	aa73590d5265031c5bb64b5c0e7f84c44cf5f8539e6d8606b763adac784e8b2e
@openzeppelin/contracts/token/ERC20/ERC20.sol	5031430cc2613c32736d598037d3075985a2a09e61592a013dbd09a5bc2041b8
@openzeppelin/contracts/token/ERC20/extensions/draft-IERC20Permit.sol	3e7aa0e0f69eec8f097ad664d525e7b3f0a3fda8dcdd97de5433ddb131db86ef
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	af5c8a77965cc82c33b7ff844deb9826166689e55dc037a7f2f790d057811990
@openzeppelin/contracts/token/ERC20/IERC20.sol	94f23e4af51a18c2269b355b8c7cf4db8003d075c9c541019eb8dcf4122864d5
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	fa36a21bd954262006d806b988e4495562e7b50420775e2aa0deecb596fd1902
@openzeppelin/contracts/token/ERC721/IERC721.sol	fde830ac73ef320f7e3ce977b8cf567173f1e479ba86d584498f8362a67a5dc0

@openzeppelin/contracts/utils/Address.sol	1e0922f6c0bf6b1b8b4d480dcabb691b1359195a297bde6dc5172e79f3a1f826
@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/utils/introspection/IERC165.sol	701e025d13ec6be09ae892eb029cd83b3064325801d73654847a5fb11c58b1e5
contracts/CompBet.sol	7d82a5cf0bd8a58345421e33575cfaabfc82f39ba826083bf4d1b4e42829b267
contracts/IABPToken.sol	b83a1d96d45720c3bb1f46b865fd4a6097748a7a671941b7c5ff02b2202c1e3e

Not Transferable NFT

The Cyberscope hereby confirms there is not function within the contract that permits the transfer of an NFT from a player to another location (wallet or contract). This contract does permit registering the owner of the asset, but does not require the transfer of any kind to an AlphaBets contract or any other wallet or location. Players can confirm when performing a ‘staking’ action that no transfer rights are required as part of the transaction.

Introduction

The CompBet contract implements a betting contract. To achieve randomness the contract utilized Chainlink VRF.

The users have the ability to place a bet from one side. The bet amount consists either from native currency or NFTs. The game has two sides. Each side increases its chance to win proportionally to the accumulated bet amount. The users that lose have the ability to claim one thousand times the equivalent amount in ABPToken.

- The ‘staked’ NFTs are not transferred to the contract. The contract merely marks that an NFT is owned by the corresponding user.
- The ABPToken is not transferable.

Roles

The ‘manager’ role plays an important role in the contract. Due to the fact that they have the authority to:

- Create new betting battles.
- Unstake NFTs from users. Since the NFTs are not transferable, that means that the NFTs tracking variables are getting erased.
- Initiate and finalise a battle.

Contract Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	SNF	Stacked NFTs Duplication	Acknowledged
●	SND	Reward Funds Calculation	Acknowledged
●	DSC	Data Structure Concern	Unresolved
●	MC	Missing Check	Unresolved
●	L02	State Variables could be Declared Constant	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L07	Missing Events Arithmetic	Unresolved
●	L14	Uninitialized Variables in Local Scope	Acknowledged

SND - Stacked NFTs Duplication

Criticality	medium
Location	contract.sol#L202
Status	Acknowledged

Description

The term ‘stake’ NFTs is missleading since there is no transferring functionality taking place. The CompBet contract merely marks that a specific NFT is owned by the owner. The contract does not track if the NFTs have already been marked by any other user. As a result we the following vulnerability can be produced:

1. User A stakes an NFT.
2. User A transfers the NFT to user B.
3. User B stakes the same NFT.
4. User B transfers the NFT to user C etc.

```
function stakeNft(
  uint256 battleId,
  uint256[] calldata tokenIds,
  bool _side
) {...}
```

Recommendation

The contract should track the NFTs that have already ‘staked’ in order to prevent potential duplications.

Updated 14 October 2022

The team has acknowledged that it is not a security issue.

Team’s Reply

This issue does not consider our off-chain monitoring solution.

Our off-chain solution monitors movement of NFT. This is done using a cronjob which watches the transfer events within the NFT Collection contract and disables this exploit at battle end. The result is that only the original NFT ‘staker’ can claim rewards within the ‘staking’ wallet.
This was performed in order to reduce tx fees on ETH which will grow exponentially with more usage. This solution involves no gas payment on both the Staker and the Company side.

SND - Reward Funds Calculation

Criticality	medium
Location	contract.sol#L240
Status	Acknowledged

Description

The claimReward method is using the getClaimableAmount to calculate the awarded amount. The getClaimableAmount is a result of total native currency that has been accumulated by the placeBet method and the quantity of ‘stacked’ NFTs. The claimReward method redeems the reward on native currency. Since the ‘staked’ NFTs are not transferring native currency to the contract, then the awarded amount may not be sufficient to cover the expense.

```
function getClaimableAmount(uint256 battleId, address _user)
public
view
battleFinalized(battleId)
returns (uint256)
{
return (getClaimableAmountForBetter(battleId, _user) +
    getClaimableAmountForNftStaker(battleId, _user));
}
```

Recommendation

The contract should not calculate the ‘staked’ NFTs as part of the native currency reward process.

Updated 14 October 2022

The team has acknowledged that it is not a security issue.

Team’s Reply

The assessment says that players can potentially exploit by staking and NOT adding ETH. But this type of gameplay is part of the design.

Per our documentation , putting ETH into the game is not a requirement for obtaining the jackpot and receiving jackpot for only staking is part of our core gameplay. This is not only allowed but also encouraged for players to participate in this way.

AlphaBets : How to Play guide
https://alphabetsgg.notion.site/alphabetsgg/How-to-Play-a21b75c6b65c4bb9828c1fd9d0962424#32c9eab68ce747f8b995f313ec6430ab

DSC - Data Structure Concern

Criticality	minor
Location	contract.sol
Status	Unresolved

Description

The contract uses a mapping data structure for every variable that is related to the battle. These maps are indexed by the battle id. We assume that there are X variables related to the data structure and there are Y battles that have taken place. That means that the space complexity of X*Y will be produced. Additionally, the readability of the structure is complicated since many mappings should be combined.

```
...
// NFT Staked INFO
mapping(uint256 => mapping(bool => uint256)) public totalNftStaked;
mapping(uint256 => address) public nftCollectionA;
mapping(uint256 => address) public nftCollectionB;

mapping(uint256 => mapping(address => mapping(bool => mapping(uint256 =>
bool))))
public userNftStakeInfo; // Battle ID -> User -> Team -> Token Id -> Staked or
not
mapping(uint256 => mapping(address => mapping(bool => uint256)))
public userNftStakeLength; // User -> Team -> number of nft staked

// BET WINNER
mapping(uint256 => bool) public winnerSet;
mapping(uint256 => bool) public winner; // FALSE -> A, TRUE -> B;
...
```

Recommendation

The contract could shape a structure that contains all the variables that are required for a specific battle. This structure could be indexed by the battle id. As a result, the space complexity will decrease to Y*1=Y. Additionally, the battle state readability will be dramatically increased.

MC - Missing Check

Criticality	medium
Location	contract.sol#L598
Status	Unresolved

Description

The contract is processing variables that have not properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues.

If the `_rakePercentage`, `_nftStakersPercentage` percentages exceed a certain amount they might produce conflict or unexpected results in the rewards.

```
function setDistributionPercentage(
    uint256 _rakePercentage,
    uint256 _nftStakersPercentage
  ) external onlyOwner {
    rakePercentage = _rakePercentage;
    nftStakersPercentage = _nftStakersPercentage;
  }
```

Recommendation

The contract should properly check the variables according to the required specifications.

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contracts/CompBet.sol#L79,76,74,85,81
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
callbackGasLimit
keyHash
vrfCoordinator
numWords
requestConfirmations
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contracts/CompBet.sol#L174,600,87,205,526,88,619,328,562,525,70,553,611,173,371,352,72,305,599,527,528,89,395
Status	Unresolved

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_user
_nftStakersPercentage
s_randomWords
_side
_betEndTime
s_requestId
_abpTokenAddress
_battleId
_betStartTime
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions.

L07 - Missing Events Arithmetic

Criticality	minor / informative
Location	contracts/CompBet.sol#L598
Status	Unresolved

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
rakePercentage = _rakePercentage
```

Recommendation

Emit an event for critical parameter changes.

L14 - Uninitialized Variables in Local Scope

Criticality	minor / informative
Location	contracts/CompBet.sol#L409
Status	Acknowledged

Description

The are variables that are defined in the local scope and are not initialized.

```
unstakeCnt
```

Recommendation

All the local scoped variables should be initialized.

Updated 14 October 2022

The team has acknowledged that it is not a security issue.

Team's Reply

The assessment claims that the unstakeCnt variable defined in the local scope is not initialized.

The unstakeCnt will get the default uint256 value which is 0 which is exact required for the logic function. By not initializing, we save gas fees.

The resolution of uint256 unstakeCnt = 0; will provide the same outcome but take more gas.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
VRFCoordinatorV2Interface	Interface			
	getRequestConfig	External		-
	requestRandomWords	External	✓	-
	createSubscription	External	✓	-
	getSubscription	External		-
	requestSubscriptionOwnerTransfer	External	✓	-
	acceptSubscriptionOwnerTransfer	External	✓	-
	addConsumer	External	✓	-
	removeConsumer	External	✓	-
	cancelSubscription	External	✓	-
VRFConsumerBaseV2	Implementation			
	<Constructor>	Public	✓	-
	fulfillRandomWords	Internal	✓	
	rawFulfillRandomWords	External	✓	-
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
ReentrancyGuard	Implementation			
	<Constructor>	Public	✓	-

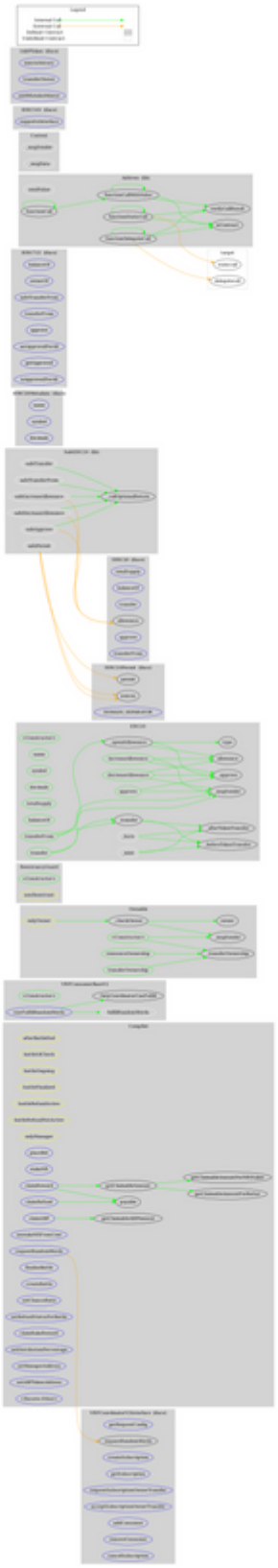
ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
IERC20Permit	Interface			
	permit	External	✓	-
	nonces	External		-
	DOMAIN_SEPARATOR	External		-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-

IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	safePermit	Internal	✓	
	_callOptionalReturn	Private	✓	
IERC721	Interface	IERC165		
	balanceOf	External		-
	ownerOf	External		-
	safeTransferFrom	External	✓	-
	safeTransferFrom	External	✓	-
	transferFrom	External	✓	-
	approve	External	✓	-
	setApprovalForAll	External	✓	-
	getApproved	External		-
	isApprovedForAll	External		-
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	

	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResult	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
IERC165	Interface			
	supportsInterface	External		-
CompBet	Implementation	Reentrancy Guard, Ownable, VRFConsumerBaseV2		
	<Constructor>	Public	✓	VRFConsumerBaseV2
	placeBet	External	Payable	nonReentrant battleIdCheck battleOngoing battleRefundNotActive
	stakeNft	External	✓	battleIdCheck battleOngoing battleRefundNotActive
	claimReward	External	✓	nonReentrant battleFinalized battleRefundNotActive
	claimABP	External	✓	nonReentrant battleFinalized battleRefundNotActive
	claimRefund	External	✓	nonReentrant battleRefundActive
	getClaimableAmountForBetter	Internal		
	getClaimableAmountForNftStaker	Internal		

	getClaimableABPAmount	Public		battleFinalized
	getClaimableAmount	Public		battleFinalized
	unstakeNftFromUser	External	✓	onlyManager
	requestRandomWords	External	✓	onlyManager afterBattleEnd battleRefundNotActive
	fulfillRandomWords	Internal	✓	
	finalizeBattle	External	✓	onlyManager afterBattleEnd
	createBattle	External	✓	onlyManager
	setChanceRatio	External	✓	onlyManager
	setRefundStatusForBattle	External	✓	battleIdCheck onlyManager
	claimRakeReward	External	✓	onlyOwner battleFinalized battleRefundNotActive
	setDistributionPercentage	External	✓	onlyOwner
	setManagerAddress	External	✓	onlyOwner
	setABPTokenAddress	External	✓	onlyOwner
	<Receive Ether>	External	Payable	-
IABPToken	Interface			
	mintArbitrary	External	✓	-
	transferOwner	External	✓	-
	addWhitelistMinter	External	✓	-

Contract Flow



Summary

The CompBet contract implements a betting mechanism. This audit mentions potential vulnerabilities, security concerns, business logic suggestions and possible optimizations.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io