



Audit Report

Bernard finance

January 2022

Audited by © coinscope

Bernard Finance Audit

CoinScope audited three sections of the Bernard finance ecosystem. Each section consists of the corresponding independent audit.

Betting Audit

<https://github.com/coinscope-co/audits/tree/main/bones/betting.pdf>

Lottery Audit

<https://github.com/coinscope-co/audits/tree/main/bones/lottery.pdf>

Staking Audit

<https://github.com/coinscope-co/audits/tree/main/bones/masterchef.pdf>



Audit Report

Bernard betting

January 2022

Type	BEP20
Network	BSC
Address	0x98Ab3bCB0E81doef369514113429Dc782C756F65
Audited by	© coinscope

Table of Contents

Table of Contents	1
Contract Review	3
Audit Updates	3
Contract Analysis	4
Contract Owner Drain Lottery Amount	4
Description	4
Recommendation	4
Exceed Limit Fees Manipulation	5
Description	5
Recommendation	5
Contract Diagnostics	6
CO - Code Optimization	7
Description	7
Recommendation	7
CR - Code Repetition	8
Description	8
Recommendation	8
L01 - Public Function could be Declared External	9
Description	9
Recommendation	9
L04 - Conformance to Solidity Naming Conventions	10
Description	10
Recommendation	10
L11 - Unnecessary Boolean equality	11
Description	11
Recommendation	11

L06 - Missing Events Access Control	12
Description	12
Recommendation	12
Contract Functions	13
Contract Flow	16
Summary	17
Disclaimer	18
About Coinscope	19

Contract Review

Contract Name	SBetting
Compiler Version	v0.6.12+commit.27d51765
Optimization	1 runs
Licence	
Explorer	https://bscscan.com/address/0x98Ab3bCB0E81daef369514113429Dc782C756F65
Source	<pre>/Users/patrick/Documents/TRON/Bernie/bones/contracts/Betting.sol, @openzeppelin/contracts/utils/ReentrancyGuard.sol, @openzeppelin/contracts/utils/Pausable.sol, @openzeppelin/contracts/utils/Context.sol, @openzeppelin/contracts/utils/Address.sol, @openzeppelin/contracts/token/ERC20/SafeERC20.sol , @openzeppelin/contracts/token/ERC20/IERC20.sol, @openzeppelin/contracts/math/SafeMath.sol, @openzeppelin/contracts/access/Ownable.sol</pre>

Audit Updates

Initial Audit	29th January 2022
Corrected	

Contract Analysis

Contract Owner Drain Lottery Amount

Criticality	medium
Location	contract.sol#L332

Description

The users that place bets are transferring tokens to the contract. These tokens can be drained by the contract owner anytime by calling the *inCaseTokensGetStuck()* function with the bones address as argument. As a result the winners may not be able to claim their rewards.

```
function inCaseTokensGetStuck(address _token,uint256 _amount) public onlyAdmin{
    require(IERC20(_token).balanceOf(address(this)) > 0, 'Error, contract has
insufficient balance');
    IERC20(_token).safeTransfer(address(adminAddress), _amount);
}
```

Recommendation

There could be some limitation regarding the contract owner's access. For instance, the contract owner should be allowed to drain tokens only on legacy bets.

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

Exceed Limit Fees Manipulation

Criticality	minor
Location	contract.sol#L133

Description

The contract owner has the authority to increase the fees. The owner may take advantage of it by calling the function *setRewardRate* with a high percentage value. The *rewardRate* amount accumulates fees that are transferred to the owner's wallet.

```
function setRewardRate(uint256 _rewardRate) external onlyAdmin {  
    require(_rewardRate <= TOTAL_RATE, "rewardRate cannot be more than 100%");  
    rewardRate = _rewardRate;  
    treasuryRate = TOTAL_RATE.sub(_rewardRate);  
  
    emit RatesUpdated(rewardRate, treasuryRate);  
}
```

```
function claimTreasury() external onlyAdmin {  
    uint256 currentTreasuryAmount = treasuryAmount;  
    treasuryAmount = 0;  
    bones.safeTransfer(address(adminAddress), currentTreasuryAmount);  
    emit ClaimTreasury(currentTreasuryAmount);  
}
```

Recommendation

The contract could embody a check for the maximum acceptable value.

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	CO	Code Optimization
●	CR	Code Repetition
●	L01	Public Function could be Declared External
●	L04	Conformance to Solidity Naming Conventions
●	L11	Unnecessary Boolean equality
●	L06	Missing Events Access Control

CO - Code Optimization

Criticality	minor
Location	contract.sol#L282

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The side number can only be a number between 0 to 3. Hence, the else if statement is not required since the only possible value is 3.

```
else if (side == 3){  
    round.refundable = true;  
    rewardBaseCalAmount = 0;  
    treasuryAmt = 0;  
    rewardAmount = 0;  
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

CR - Code Repetition

Criticality	minor
Location	contract.sol#L269

Description

There are code segments that are repetitive in the contract. Those segments increase the code size of the contract unnecessarily.

```
if (side == 0) {
    rewardBaseCalAmount = round.homeAmount;
    rewardAmount = round.totalAmount.mul(rewardRate).div(TOTAL_RATE);
    treasuryAmt = round.totalAmount.mul(treasuryRate).div(TOTAL_RATE);
}
else if (side == 1 ) {
    rewardBaseCalAmount = round.drawAmount;
    rewardAmount = round.totalAmount.mul(rewardRate).div(TOTAL_RATE);
    treasuryAmt = round.totalAmount.mul(treasuryRate).div(TOTAL_RATE);
}
else if (side == 2 ) {
    rewardBaseCalAmount = round.awayAmount;
    rewardAmount = round.totalAmount.mul(rewardRate).div(TOTAL_RATE);
    treasuryAmt = round.totalAmount.mul(treasuryRate).div(TOTAL_RATE);
}
```

Recommendation

Create an internal function that contains the code segment and remove it from all the sections.

L01 - Public Function could be Declared External

Criticality

minor

Location

contracts/Betting.sol#L332,L163,L85 and 1 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
inCaseTokensGetStuck  
bet  
unpause  
...
```

Recommendation

Use the external attribute for functions never called from the contract

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contracts/Betting.sol#L332,L149,L141 and 4 more

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.

```
_amount  
_token  
_minBetAmount  
...
```

Recommendation

Follow the Solidity naming convention.

<https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions>

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contracts/Betting.sol#L298,L246,L232 and 1 more

Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
require(bool,string)(rounds[epoch].oracleCalled == true,game not ended)
require(bool,string)(rounds[epoch].oracleCalled == false,game ended)
rounds[epoch].startBlock != 0 && block.number > rounds[epoch].startBlock &&
rounds[epoch].oracleCalled == false && block.timestamp < rounds[epoch].endEpoch
...
```

Recommendation

Remove the equality to the boolean constant.

L06 - Missing Events Access Control

Criticality

minor

Location

contracts/Betting.sol#L128,L119

Description

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

```
operatorAddress = _operatorAddress  
adminAddress = _adminAddress
```

Recommendation

Emit an event for critical parameter changes.

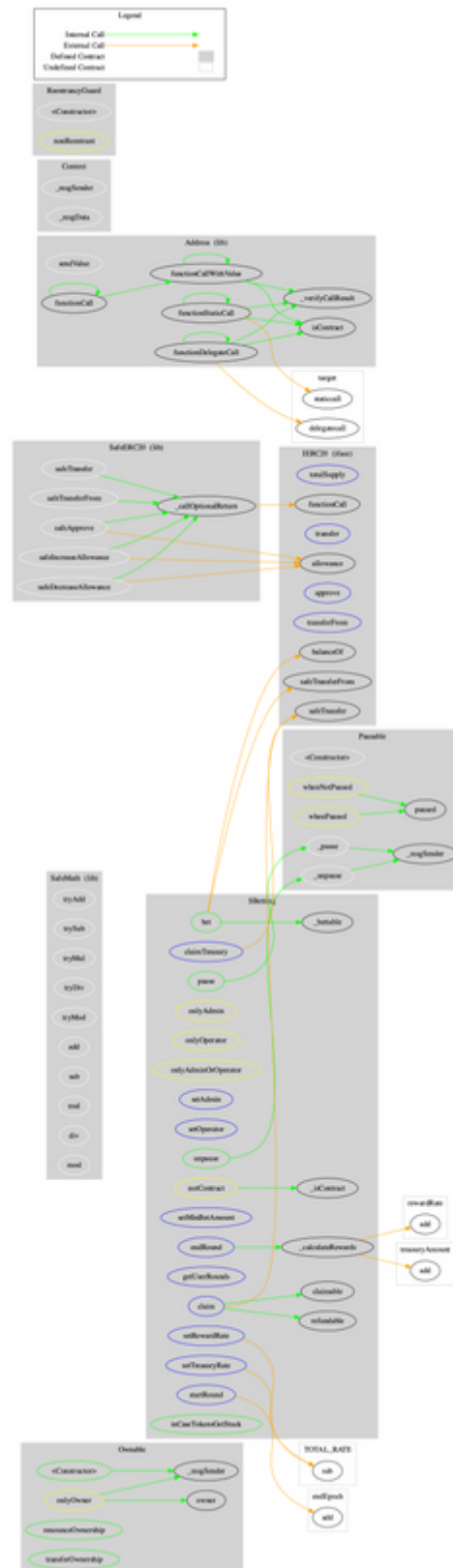
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Ownable	Implementation	Context		
	<Constructor>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
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	✓	
	_callOptionalReturn	Private	✓	
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	_verifyCallResult	Private		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Pausable	Implementation	Context		
	<Constructor>	Internal	✓	
	paused	Public		-
	_pause	Internal	✓	whenNotPaused
	_unpause	Internal	✓	whenPaused
ReentrancyGuard	Implementation			
	<Constructor>	Internal	✓	

S Betting	Implementation	Ownable, Reentrancy Guard, Pausable		
	<Constructor>	Public	✓	-
	pause	Public	✓	onlyAdminOrOperator whenNotPaused
	unpause	Public	✓	onlyAdmin whenPaused
	_isContract	Internal		
	setAdmin	External	✓	onlyOwner
	setOperator	External	✓	onlyOwner
	setRewardRate	External	✓	onlyAdmin
	setTreasuryRate	External	✓	onlyAdmin
	setMinBetAmount	External	✓	onlyAdmin
	_bettable	Internal		
	bet	Public	Payable	nonReentrant notContract whenNotPaused
	getUserRounds	External		-
	claimable	Public		-
	refundable	Public		-
	endRound	External	✓	onlyAdminOrOperator whenNotPaused
	startRound	External	✓	onlyAdminOrOperator whenNotPaused
	_calculateRewards	Internal	✓	
	claim	External	✓	nonReentrant notContract
	claimTreasury	External	✓	onlyAdmin
	inCaseTokensGetStuck	Public	✓	onlyAdmin

Contract Flow



Summary

Bernard betting is a betting application that is built in the BSC chain. The basic logic is that users are able to bet on three possible values on every round. The admin defines the winning value for the current round. The winners can claim their rewards that is the proportional amount from all the round's bets. There are some functions that can be abused by the contract owner. We state that the owner privileges are necessary and required for proper operation of the betting application. Thus, we emphasise the contract owner to be extra careful with the credentials.

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.

Coinscope 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 Coinscope 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 Coinscope 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 Coinscope team disclaims any liability for the resulting losses.

About Coinscope

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.

Coinscope is aiming to make crypto discoverable and efficient globally. It provides all the essential tools to assist users draw their own conclusions.



The Coinscope.co team

<https://www.coinscope.co>



Audit Report

Bernard lottery

January 2022

Type BEP20

Network BSC

Address 0x189396f720a3C31E7650cC3eD066797B4D7fC0CF

Audited by © coinscope

Table of Contents

Table of Contents	1
Contract Review	3
Audit Updates	3
Contract Analysis	4
Contract Owner Drain Lottery Amount	4
Description	4
Recommendation	4
Contract Diagnostics	5
CR - Code Repetition	6
Description	6
Recommendation	6
L01 - Public Function could be Declared External	7
Description	7
Recommendation	7
L04 - Conformance to Solidity Naming Conventions	8
Description	8
Recommendation	8
L09 - Dead Code Elimination	9
Description	9
Recommendation	9
L07 - Missing Events Arithmetic	10
Description	10
Recommendation	10
L06 - Missing Events Access Control	11
Description	11
Recommendation	11

Contract Functions	12
Contract Flow	19
Summary	20
Disclaimer	21
About Coinscope	22

Contract Review

Contract Name	Lottery
Compiler Version	v0.6.12+commit.27d51765
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/address/0x189396f720a3C31E7650cC3eD066797B4D7fC0CF
Source	contract.sol

Audit Updates

Initial Audit	30th January 2022
Corrected	

Contract Analysis

Contract Owner Drain Lottery Amount

Criticality	medium
Location	contract.sol#L332

Description

The users that buy lottery tickets are transferring tokens to the contract. These tokens can be drained by the contract owner anytime by calling the *adminWithdraw()* function. As a result the winners may not be able to claim their rewards.

```
// Withdraw without caring about rewards. EMERGENCY ONLY.  
function adminWithdraw(uint256 _amount) public onlyAdmin {  
    bern.safeTransfer(msg.sender, _amount);  
    emit DevWithdraw(msg.sender, _amount);  
}
```

Recommendation

There could be some limitation regarding the contract owner's access. For instance, the contract owner should be allowed to drain tokens only on legacy lotteries.

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	CR	Code Repetition
●	L01	Public Function could be Declared External
●	L04	Conformance to Solidity Naming Conventions
●	L09	Dead Code Elimination
●	L07	Missing Events Arithmetic
●	L06	Missing Events Access Control

CR - Code Repetition

Criticality	minor
Location	contract.sol#L1

Description

There are code segments that are repetitive in the contract. Those segments increase the code size of the contract unnecessarily.

The functions *buy()* and *multiBuy()* share the same statements internally.

```
for (uint i = 0; i < 4; i++) {
    require (_numbers[i] <= maxNumber, 'exceed number scope');
}
uint256 tokenId = lotteryNFT.newLotteryItem(msg.sender, _numbers, _price,
issueIndex);
lotteryInfo[issueIndex].push(tokenId);
if (userInfo[msg.sender].length == 0) {
    totalAddresses = totalAddresses + 1;
}
userInfo[msg.sender].push(tokenId);
totalAmount = totalAmount.add(_price);
lastTimestamp = block.timestamp;
uint64[keyLengthForEachBuy] memory userNumberIndex =
generateNumberIndexKey(_numbers);
for (uint i = 0; i < keyLengthForEachBuy; i++) {

userBuyAmountSum[issueIndex][userNumberIndex[i]]=userBuyAmountSum[issueIndex][us
erNumberIndex[i]].add(_price);
}
```

Recommendation

Create an internal function that contains the code segment and remove it from all the sections.

L01 - Public Function could be Declared External

Criticality

minor

Location

contract.sol#L2609,L2604,L2599 and 19 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
adminWithdraw  
setLotteryNFT  
setAdmin  
...
```

Recommendation

Use the external attribute for functions never called from the contract

L04 - Conformance to Solidity Naming Conventions

Criticality

minor

Location

contract.sol#L2249,L2625,L2620 and 30 more

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.

```
keyLengthForEachBuy  
_allcation3  
_allcation2  
...
```

Recommendation

Follow the Solidity naming convention.

<https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions>

L09 - Dead Code Elimination

Criticality

minor

Location

contract.sol#L352,L362,L387 and 37 more

Description

Functions that are not used in the contract, and make the code's size bigger.

```
trySub  
tryMul  
tryMod  
...
```

Recommendation

Remove unused functions.

L07 - Missing Events Arithmetic

Criticality

minor

Location

contract.sol#L2620,L2615

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.

```
maxNumber = _maxNumber  
minPrice = _price
```

Recommendation

Emit an event for critical parameter changes.

L06 - Missing Events Access Control

Criticality

minor

Location

contract.sol#L2599

Description

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

```
adminAddress = _adminAddress
```

Recommendation

Emit an event for critical parameter changes.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
IERC165	Interface			
	supportsInterface	External		-
IERC721	Interface	IERC165		
	balanceOf	External		-
	ownerOf	External		-
	safeTransferFrom	External	✓	-
	transferFrom	External	✓	-
	approve	External	✓	-
	getApproved	External		-
	setApprovalForAll	External	✓	-
	isApprovedForAll	External		-
	safeTransferFrom	External	✓	-
IERC721Metadata	Interface	IERC721		
	name	External		-
	symbol	External		-
	tokenURI	External		-
IERC721Enumerable	Interface	IERC721		
	totalSupply	External		-
	tokenOfOwnerByIndex	External		-
	tokenByIndex	External		-

IERC721Receiver	Interface			
	onERC721Received	External	✓	-
ERC165	Implementation	IERC165		
	<Constructor>	Internal	✓	
	supportsInterface	Public		-
	_registerInterface	Internal	✓	
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	

	_verifyCallResult	Private		
EnumerableSet	Library			
	_add	Private	✓	
	_remove	Private	✓	
	_contains	Private		
	_length	Private		
	_at	Private		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
EnumerableMap	Library			
	_set	Private	✓	
	_remove	Private	✓	
	_contains	Private		
	_length	Private		
	_at	Private		
	_tryGet	Private		
	_get	Private		
	_get	Private		
	set	Internal	✓	

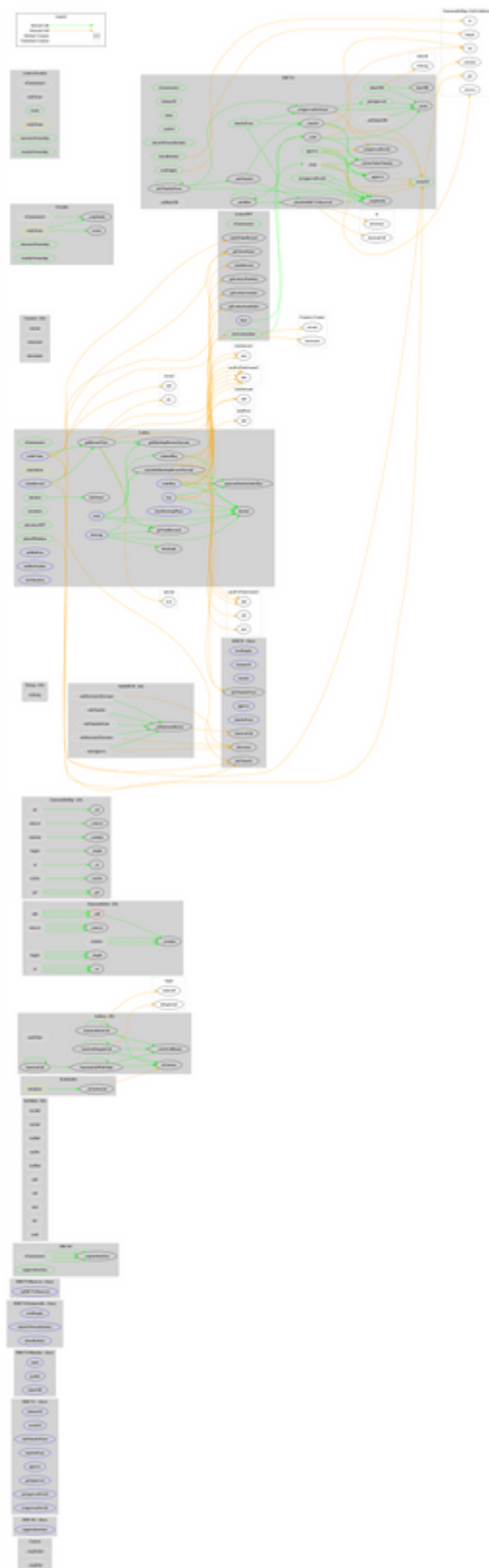
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	tryGet	Internal		
	get	Internal		
	get	Internal		
Strings	Library			
	toString	Internal		
ERC721	Implementation	Context, ERC165, IERC721, IERC721Me tadata, IERC721En umerable		
	<Constructor>	Public	✓	-
	balanceOf	Public		-
	ownerOf	Public		-
	name	Public		-
	symbol	Public		-
	tokenURI	Public		-
	baseURI	Public		-
	tokenOfOwnerByIndex	Public		-
	totalSupply	Public		-
	tokenByIndex	Public		-
	approve	Public	✓	-
	getApproved	Public		-
	setApprovalForAll	Public	✓	-
	isApprovedForAll	Public		-
	transferFrom	Public	✓	-
	safeTransferFrom	Public	✓	-
	safeTransferFrom	Public	✓	-
	_safeTransfer	Internal	✓	
	_exists	Internal		

	_isApprovedOrOwner	Internal		
	_safeMint	Internal	✓	
	_safeMint	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_transfer	Internal	✓	
	_setTokenURI	Internal	✓	
	_setBaseURI	Internal	✓	
	_checkOnERC721Received	Private	✓	
	_approve	Private	✓	
	_beforeTokenTransfer	Internal	✓	
Counters	Library			
	current	Internal		
	increment	Internal	✓	
	decrement	Internal	✓	
Ownable	Implementation	Context		
	<Constructor>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
LotteryNFT	Implementation	ERC721, Ownable		
	<Constructor>	Public	✓	ERC721
	newLotteryItem	Public	✓	onlyOwner
	getLotteryNumbers	External		-
	getLotteryAmount	External		-
	getLotteryIssueIndex	External		-
	claimReward	External	✓	onlyOwner
	multiClaimReward	External	✓	onlyOwner
	burn	External	✓	onlyOwner
	getClaimStatus	External		-
LotteryOwnabl	Implementation			

e				
	<Constructor>	Internal	✓	
	initOwner	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
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	✓	
	_callOptionalReturn	Private	✓	
Initializable	Implementation			
	_isConstructor	Private		
Lottery	Implementation	LotteryOwn able, Initializable		
	<Constructor>	Public	✓	-
	initialize	Public	✓	initializer
	drawed	Public		-
	reset	External	✓	onlyAdmin
	enterDrawingPhase	External	✓	onlyAdmin
	drawing	External	✓	onlyAdmin
	internalBuy	Internal	✓	

	buy	External	✓	-
	multiBuy	External	✓	-
	claimReward	External	✓	-
	multiClaim	External	✓	-
	generateNumberIndexKey	Public		-
	calculateMatchingRewardAmount	Internal		
	getMatchingRewardAmount	Public		-
	getTotalRewards	Public		-
	getRewardView	Public		-
	setAdmin	Public	✓	onlyOwner
	setLotteryNFT	Public	✓	onlyOwner
	adminWithdraw	Public	✓	onlyAdmin
	setMinPrice	External	✓	onlyAdmin
	setMaxNumber	External	✓	onlyAdmin
	setAllocation	External	✓	onlyAdmin

Contract Flow



Summary

Bernard lottery is a lottery application that is built in the BSC chain. The lottery is proceeding in rounds. Users are buying their lottery tickets. In the lottery ticket, the users choose the amount that they want to supply and the numbers that they are guessing. Then, the contract owner draws the random numbers. The users that guess correctly can claim their proportional rewards.

Even though the blockchain is a deterministic environment, the randomization function that is setting the winning numbers is well implemented. The algorithm combines fee costs, block information and an external number. This combination produces quite random numbers and it is very difficult to be manipulated.

There are some functions that can be abused by the contract owner. We state that the owner privileges are necessary and required for proper operation of the lottery application. Thus, we emphasise the contract owner to be extra careful with the credentials.

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.

Coinscope 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 Coinscope 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 Coinscope 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 Coinscope team disclaims any liability for the resulting losses.

About Coinscope

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.

Coinscope is aiming to make crypto discoverable and efficient globally. It provides all the essential tools to assist users draw their own conclusions.



The Coinscope.co team

<https://www.coinscope.co>



Audit Report

Bernard masterchef

January 2022

Type	BEP20
Network	BSC
Address	0xCC097123af369385e13e1dfea18cD334f1D8273E
Audited by	© coinscope

Table of Contents

Table of Contents	1
Contract Review	3
Audit Updates	3
Contract Analysis	4
Exceed Limit Fees Manipulation	4
Description	4
Recommendation	4
NFT Levels Manipulation	5
Description	5
Recommendation	5
Contract Diagnostics	6
L01 - Public Function could be Declared External	7
Description	7
Recommendation	7
L04 - Conformance to Solidity Naming Conventions	8
Description	8
Recommendation	8
L09 - Dead Code Elimination	9
Description	9
Recommendation	9
L07 - Missing Events Arithmetic	10
Description	10
Recommendation	10
Contract Functions	11
Contract Flow	15
Summary	16

Disclaimer**17****About Coinscope****18**

Contract Review

Contract Name	ChefBernard
Compiler Version	v0.6.12+commit.27d51765
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/address/0xCC097123af369385e13e1dfea18cD334f1D8273E
Source	contract.sol

Audit Updates

Initial Audit	30th January 2022
Corrected	

Contract Analysis

Exceed Limit Fees Manipulation

Criticality	minor
Location	contract.sol#L133

Description

The contract owner has the authority to increase the fees. The owner may take advantage of it by calling the function *updateTaxedContract* with a high percentage value. The rewardRate amount accumulates fees that are transferred to the owner's wallet.

```
function updateTaxedContract(address _addr, bool flag, uint256 _taxeFee) public  
onlyOwner {  
    taxedContract[_addr] = flag;  
    taxeFee = _taxeFee;  
}
```

Recommendation

The contract could embody a check for the maximum acceptable value.

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

NFT Levels Manipulation

Criticality	minor
Location	contract.sol#L133

Description

The contract owner has the authority to change the level of an NFT. The owner may take advantage of it by calling the function *updateNftInfo* with a high percentage value. The NFT level is used in order to allow a user to claim his rewards.

```
function updateNftInfo(address _nftAddress, uint _tokenId, uint8 _newlevel)
public {
    require(nftManager == msg.sender, "nftManager: not permitted");
    require(_newlevel > 0, "nftManager: invalid level");
    require(_newlevel < 4, "nftManager: invalid level");
    require(nftLevel[_nftAddress][_tokenId] != _newlevel, "nftManager: invalid
level");
    nftLevel[_nftAddress][_tokenId] = _newlevel;
}
```

```
if( user.userLevel < pool.poolLevel) {
    pending = 0;
}

if(pending > 0) {
    safeBONESTransfer(msg.sender, pending);
}
```

Recommendation

The contract could embody a check for not allowing mutating the NFT level, when the holder of the NFT is participating in a pool.

The owner should carefully manage the credentials of the owner's account. We advised considering an extra-strong security mechanism that the actions may be quarantined by many users instead of one.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	L01	Public Function could be Declared External
●	L04	Conformance to Solidity Naming Conventions
●	L09	Dead Code Elimination
●	L07	Missing Events Arithmetic

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L1304,L1299,L1294 and 22 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
withdrawBEP20  
updateTaxedContract  
updateEmissionRate  
...
```

Recommendation

Use the external attribute for functions never called from the contract

L04 - Conformance to Solidity Naming Conventions

Criticality

minor

Location

contract.sol#L1052,L1050,L1048 and 36 more

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.

```
BONUS_MULTIPLIER  
BONESPerBlock  
BONES  
...
```

Recommendation

Follow the Solidity naming convention.

<https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions>

L09 - Dead Code Elimination

Criticality

minor

Location

contract.sol#L41,L51,L76 and 17 more

Description

Functions that are not used in the contract, and make the code's size bigger.

```
trySub  
tryMul  
tryMod  
...
```

Recommendation

Remove unused functions.

L07 - Missing Events Arithmetic

Criticality

minor

Location

contract.sol#L1294,L1118,L1101 and 1 more

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.

```
BONESPerBlock = _BONESPerBlock
totalAllocPoint = totalAllocPoint.sub(prevAllocPoint).add(_allocPoint)
totalAllocPoint = totalAllocPoint.add(_allocPoint)
...
```

Recommendation

Emit an event for critical parameter changes.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
IBEP20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Address	Library			
	isContract	Internal		

	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	_verifyCallResult	Private		
SafeBEP20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	_callOptionalReturn	Private	✓	
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
BEP20	Implementation	Context, IBEP20, Ownable		
	<Constructor>	Public	✓	-
	getOwner	External		-
	name	Public		-
	symbol	Public		-

	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	mint	Public	✓	onlyOwner
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_burnFrom	Internal	✓	
IMigratorChef	Interface			
	migrate	External	✓	-
IERC165	Interface			
	supportsInterface	External		-
IERC721	Interface	IERC165		
	totalSupply	External		-
	balanceOf	External		-
	ownerOf	External		-
	tokenOfOwnerByIndex	External		-
ChefBernard	Implementation	Ownable		
	<Constructor>	Public	✓	-
	updateMultiplier	Public	✓	onlyOwner
	poolLength	External		-
	add	Public	✓	onlyOwner
	set	Public	✓	onlyOwner
	getMultiplier	Public		-

	pendingBONES	External		-
	massUpdatePools	Public	✓	-
	updatePool	Public	✓	-
	deposit	Public	✓	-
	withdraw	Public	✓	-
	emergencyWithdraw	Public	✓	-
	safeBONESTransfer	Internal	✓	
	addNftInfo	Public	✓	nonDuplicated
	updateNftInfo	Public	✓	-
	setNftManager	Public	✓	-
	getUserLevel	Public		-
	updateEmissionRate	Public	✓	onlyOwner
	updateTaxedContract	Public	✓	onlyOwner
	withdrawBEP20	Public	✓	onlyOwner

Contract Flow



Summary

Bernard masterchef is a stacking application. Users can deposit lp tokens in order to get BONES tokens as a reward. The award is proportional to the time period that has elapsed and the deposited amount. Users can claim their rewards any time. There are pools that require the users to hold NFTs with specific “level”. The level of the NFT is defined by the admins. There are some functions that can be abused by the contract owner. We state that the owner privileges are necessary and required for proper operation of the betting application. Thus, we emphasise the contract owner to be extra careful with the credentials.

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.

Coinscope 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 Coinscope 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 Coinscope 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 Coinscope team disclaims any liability for the resulting losses.

About Coinscope

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.

Coinscope is aiming to make crypto discoverable and efficient globally. It provides all the essential tools to assist users draw their own conclusions.



The Coinscope.co team

<https://www.coinscope.co>