Audit Report

Bernard betting

January 2022

Type	BEP20
Network	BSC
Address	0x98Ab3bCB0E81daef369514113429Dc782C756F65
Audited by	© coinscope

Table of Contents

Contract Review

Contract Name	SBetting
Compiler Version	v0.6.12+commit.27d51765
Optimization	1 runs
Licence	
Explorer	https://bscscan.com/address/0x98Ab3bCB0E81daef369514113429Dc782C756F65
Source	/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

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
insufficent 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.

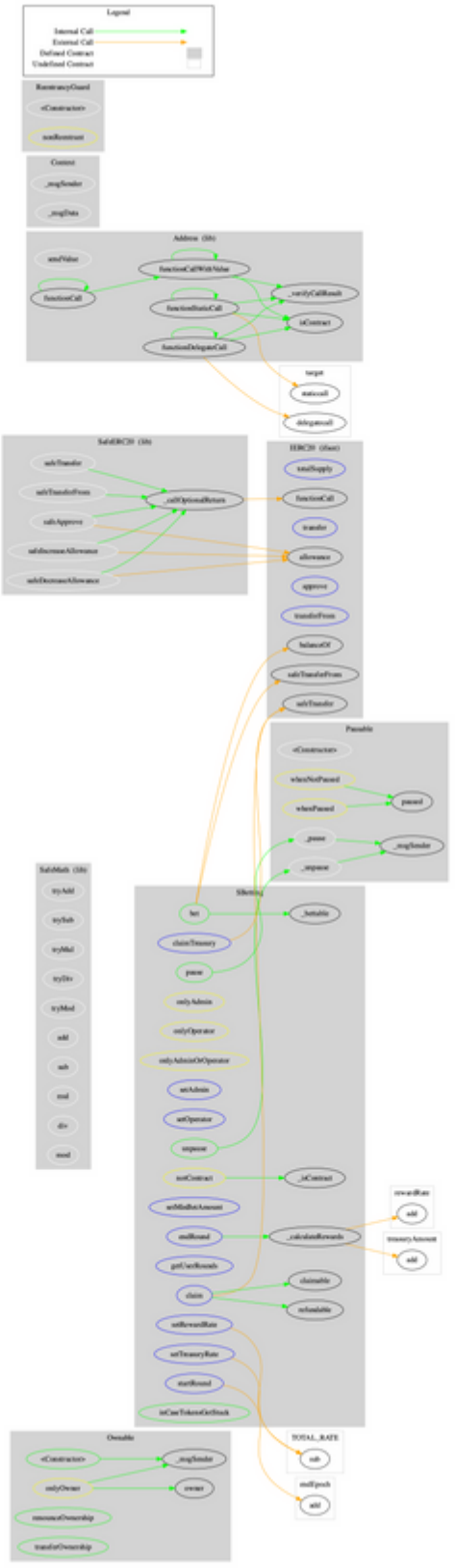
Contract Functions

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	✓	

SBetting	Implementation	Ownable, ReentrancyGuard, 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