Smart Contract Security Audit

Audit details:

Audited project:	CHANGPENGUIN
Deployer address	0x5A979125f0330C603BF56f097eD00E982147149A
Blockchain:	Binance Smart Chain
Project website:	https://changpenguin.finance/

May, 2021

TechRate

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (TechRate) owe no duty of care towards you or any other person, nor does TechRate make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and TechRate hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, TechRate hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against TechRate, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

TechRate was commissioned by CHANGPENGUIN to perform an audit of smart contracts:

https://bscscan.com/token/0x5A979125f0330C603BF56f097eD00E982147149A

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts details

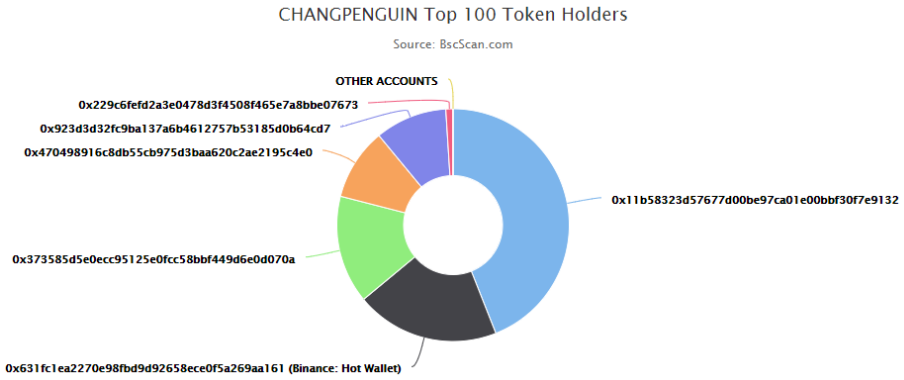
Token contract details for 01.05.2021.

Contract name:	CHANGPENGUIN
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0x5A979125f0330C603BF56f097eD00E982147149A
Total supply:	100_000_000_000_000_000_000
Token ticker:	CHANGPENGUIN
Decimals:	18
Token holders:	463
Transactions count:	2191
Top 100 holders dominance:	95.11 %
Contract deployer address:	0x11B58323D57677D00bE97CA01E00bbF30f7e9132
Contract's current owner address:	0x11B58323D57677D00bE97CA01E00bbF30f7e9132

CHANGPENGUIN top 10 token holders

Rank	Address	Quantity	Percentage	Analytics
1	0x11b58323d57677d00be97ca01e00bbf30f7e9132	44,000,000,000,000	44.0000%	
2	Binance: Hot Wallet	20,000,000,000,000	20.0000%	
3	0x373585d5e0ecc95125e0fcc58bbf449d6e0d070a	15,000,000,000,000	15.0000%	
4	0x470498916c8db55cb975d3baa620c2ae2195c4e0	10,000,000,000,000	10.0000%	
5	0x923d3d32fc9ba137a6b4612757b53185d0b64cd7	10,000,000,000,000	10.0000%	
6	0x229c6fefd2a3e0478d3f4508f465e7a8bbe07673	1,000,000,000,000	1.0000%	

CHANGPENGUIN top 10 token distribution



CHANGPENGUIN contract interaction details

Masterchef contract details for 01.05.2021.

Contract name:	MasterChef
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0x5A979125f0330C603BF56f097eD00E982147149A
Dev address:	0x11B58323D57677D00bE97CA01E00bbF30f7e9132
Fee address:	0xdbec8165bc99ca14c54281029a2505551fc5940a
Token contract address:	0x5A979125f0330C603BF56f097eD00E982147149A
Token per block:	1_000_000_000_000_000_000
Contract owner address:	0x402d745c21a792dae1de4d38594f3b084d049b10
Pool length:	27
Start block:	7109999
Total alloc point:	13400
Bonus multiplier:	1
Referral commission rate:	200
Referral contract address:	0x43ee4a63720b3d114638ae7678ac405f8fafd578

MasterChef contract Pools info:

Pool with id 0:

lpToken *address* : 0x11b58323d57677d00be97ca01e00bbf30f7e9132

allocPoint *uint256* : 4000
lastRewardBlock *uint256* : 7109999
accCHANGPENGUINPerShare *uint256*
: 0 depositFeeBP *uint16* : 0

Pool with id 1:

lpToken *address* : 0x631fc1ea2270e98fbd9d92658ece0f5a269aa161
allocPoint *uint256* : 2400
lastRewardBlock *uint256* : 7109999
accCHANGPENGUINPerShare *uint256*
: 0 depositFeeBP *uint16* : 0

Pool with id 2:

lpToken *address* : 0x373585d5e0ecc95125e0fcc58bbf449d6e0d070a

allocPoint *uint256* : 500
lastRewardBlock *uint256* : 7109999
accCHANGPENGUINPerShare *uint256*
: 0 depositFeeBP *uint16* : 400

Pool with id 3:

lpToken *address* : 0x470498916c8db55cb975d3baa620c2ae2195c4e0
allocPoint *uint256* : 400
lastRewardBlock *uint256* : 7109999
accCHANGPENGUINPerShare *uint256*
: 0 depositFeeBP *uint16* : 400

Pool with id 4:

lpToken *address* : 0x923d3d32fc9ba137a6b4612757b53185d0b64cd7
allocPoint *uint256* : 600
lastRewardBlock *uint256* : 7109999
accCHANGPENGUINPerShare *uint256*
: 0 depositFeeBP *uint16* : 400

Pool with id 5:

lpToken *address* : 0x229c6fefd2a3e0478d3f4508f465e7a8bbe07673allocPoint
uint256 : 600
lastRewardBlock *uint256* : 7109999

accCHANGPENGUINPerShare *uint256*
: 0 depositFeeBP *uint16* : 400

Issues Checking Status

№	Issue description.	Checking status
1	Compiler errors.	Passed
2	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Low issues
10	Methods execution permissions.	Passed
11	Economy model of the contract.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Medium issues
19	Cross-function race conditions.	Passed
20	Safe Open Zeppelin contracts implementation and usage.	Passed
21	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

1. Wrong burning

Issue:

There is sending tokens to the dead address in overridden _transfer functions, instead of burning them in token contract.

```
function _transfer(address sender↑, address recipient↑, uint256 amount↑) internal virtual override {
    if (recipient↑ == BURN_ADDRESS) {
        super._transfer(sender↑, recipient↑, amount↑);
    } else {
        // 2% of every transfer burnt
        uint256 burnAmount = amount↑.mul(2).div(100);
        // 98% of transfer sent to recipient
        uint256 sendAmount = amount↑.sub(burnAmount);
        require(amount↑ == sendAmount + burnAmount, "JAGUAR::transfer: Burn value invalid");

        super._transfer(sender↑, BURN_ADDRESS, burnAmount);
        super._transfer(sender↑, recipient↑, sendAmount);
        amount↑ = sendAmount;
    }
}
```

Recommendation:

There should be a burn instead of sending to the dead address.

Low Severity Issues

1. Block gas limit

Issue:

The updateEmissionRate function can fail due to block gas limit if the pool size is too big.

```
function updateEmissionRate() public {
    require(block.number > startBlock, "updateEmissionRate: Can only be called after mining starts");
    require(jaguarPerBlock > MINIMUM_EMISSION_RATE, "updateEmissionRate: Emission rate has reached the minimum threshold");

    uint256 currentIndex = block.number.sub(startBlock).div(EMISSION_REDUCTION_PERIOD_BLOCKS);
    if (currentIndex <= lastReductionPeriodIndex) {
        return;
    }

    uint256 newEmissionRate = jaguarPerBlock;
    for (uint256 index = lastReductionPeriodIndex; index < currentIndex; ++index) {
        newEmissionRate = newEmissionRate.mul(1e4 - EMISSION_REDUCTION_RATE_PER_PERIOD).div(1e4);
    }

    newEmissionRate = newEmissionRate < MINIMUM_EMISSION_RATE ? MINIMUM_EMISSION_RATE : newEmissionRate;
    if (newEmissionRate >= jaguarPerBlock) {
        return;
    }

    massUpdatePools();
    lastReductionPeriodIndex = currentIndex;
    uint256 previousEmissionRate = jaguarPerBlock;
    jaguarPerBlock = newEmissionRate;
    emit EmissionRateUpdated(msg.sender, previousEmissionRate, newEmissionRate);
}
```

2. add function issue

Issue:

If some LP token is added to the contract twice using function add, then the total amount of reward CHANGPENGUIN Reward in function updatePool will be incorrect.

```
function add(uint256 _allocPoint⬆, IBEP20 _lpToken⬆, uint16 _depositFeeBP⬆, bool _withUpdate⬆) public onlyOwner {
    require(_depositFeeBP⬆ <= 10000, "add: invalid deposit fee basis points");
    if (_withUpdate⬆) {
        massUpdatePools();
    }
    uint256 lastRewardBlock = block.number > startBlock ? block.number : startBlock;
    totalAllocPoint = totalAllocPoint.add(_allocPoint⬆);
    poolInfo.push(PoolInfo({
        lpToken: _lpToken⬆,
        allocPoint: _allocPoint⬆,
        lastRewardBlock: lastRewardBlock,
        accJaguarPerShare: 0,
        depositFeeBP: _depositFeeBP⬆
    }));
}
```

Recommendation:

Add the mapping from address to bool and check that same address will not be added twice.

Owner privileges

- ❏ Owner can withdraw tokens sent by mistake from the Referral contract.
- ❏ Owner can change the operator of the Referral contract and record Referral.
- ❏ Owner can change the jaguar referral.

Conclusion

Smart contracts contain medium severity and low severity issues.

Techrate note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.