RugFreeCoins Audit

Auto Cash Genie Token

Smart Contract Security Audit

June 13, 2022

Contents

Audit details

Audited project
Auto Cash Genie Token

Contract Address

0xf40fd8f8558ae03e2830bbf5d3e12ab86ee2a8eb
0x9c44263D12137931974f50421A46fc270dcF5566

Client contact

Auto Cash Genie Team

Blockchain

Binance smart chain

Project website

https://www.autocashgenie.finance/

Special notes

Auto Cash Genie is a proxy contract

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 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 Rugfreecoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Rugfreecoins) owe no duty of care towards you or any other person, nor does Rugfreecoins 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 Rugfreecoins 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, Rugfreecoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Rugfreecoins, 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

Rugfreecoins was commissioned by the Auto Cash Genie Team to perform an audit of the smart contract.

https://bscscan.com/token/0xf40fd8f8558ae03e2830bbf5d3e12ab86ee2a8eb

The focus of this audit is to verify that the smart contract is secure, resilient, and working according to the specifications.

The information in this report should be used to understand the risk exposure of the smart contract, project feasibility, and long-term sustainability, and as a guide to improving the security posture of the smart contract by remediating the issues that were identified.

About the project

Auto Cash Genie is a token built on the Binance Smart Chain that is with an innovative investment use case the main purpose of which is to seek out constant revenue sources. The Genie network has ability to sustainably pay returns of 2% per day and is the only deflationary daily ROI token that pays participants and referrals with a transaction tax. The recommended exchange for trading Genie is The Magic Lamp contract which can be found directly on the platforms website under the "swap" tab, as it allows us to waive the initial 10% tax on buys and provides the lowest prices and highest liquidity, resulting in less slippage for larger trades

Roadmap

- Partnership with SafuTitano
- Website Development
- Deploy Contract
- Audit Contract
- Coingecko Listing
- Coin Marketcap Listing
- Social media marketing
- Youtube marketing
- Influencer marketing

Tokenomics

10% fee when buying & selling

- 6% of trade goes to the Wishes contract in tokens
- 2% of trade goes for the marketing in tokens
- 2% of trade goes for Shore LP farms in tokens.

Target market and the concept

Target market

- Anyone who’s interested in the Crypto space with long-term investment plans.
- Anyone who’s ready to earn a passive income by holding tokens.
- Anyone who’s interested in trading tokens.
- Anyone who’s ready in receiving daily BUSD rewards of 2% from invested value.
- Anyone who’s interested in having referral rewards.
- Anyone who’s interested in taking part in the future plans of the Auto Cash Genie token.
- Anyone who’s interested in making financial transactions with any other party using Auto Cash Genie Token as the currency.

Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	10/10
3	Information quality	10/10
4	Service quality	10/10
5	System quality	9/10
6	Impact on the community	9/10
7	Impact on the business	10/10
8	Preparing for the future	9/10
9	Smart contract security	7/10
10	Smart contract functionality assessment	9/10
Total Points		**8.7/10**

Contract details

Token contract details for 13th June 2022

Contract name	Auto Cash Genie
Contract address	0xf40fD8f8558AE03e2830BBf5d3E12ab86EE2A8EB
Proxy contract connected	0x9c44263D12137931974f50421A46fc270dcF5566
Token supply	1,000,000
Token ticker	Genie
Decimals	18
Token holders	2
Transaction count	3
Marketing address	0x9bf0cc4d4b2ebf5031ace8f4f6c66e69879bed9f
Staking address	0xe1573cb20db401cd41356f01588b78a878cb259e
Vault address	0x9b30283777262e27e1389fc99e4a79a957e44052
Contract's current owner address	0xcb38b65cbb8046bae4e12b4fc302883782084c66

Contract code function details

No	Category	Item	Result
1	Coding conventions	BRC20 Token standards	pass
		compile errors	pass
		Compiler version security	pass
		visibility specifiers	pass
		Gas consumption	pass
		SafeMath features	pass
		Fallback usage	pass
		tx.origin usage	pass
		deprecated items	pass
		Redundant code	pass
		Overriding variables	pass
2	Function call audit	Authorization of function call	pass
		Low level function (call/delegate call) security	pass
		Returned value security	pass
		Self-destruct function security	pass
3	Business security	Access control of owners	Centralization issue
		Business logics	pass
		Business implementations	pass
4	Integer overflow/underflow		pass
5	Reentrancy		pass
6	Exceptional reachable state		pass
7	Transaction ordering dependence		pass
8	Block properties dependence		pass
9	Pseudo random number generator (PRNG)		pass
10	DoS (Denial of Service)		pass

11	**Token vesting implementation**		**pass**
12	**Fake deposit**		**pass**
13	**Event security**		**pass**

Contract description table

The below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions, and implementations with their visibility and mutability.

Contract	**Type**	**Bases**		
L	Function Name	Visibility	Mutability	Modifiers
Whitelist	**Implementation**	**Ownable Upgradeable**		
L	addAddressToWhitelist	Public ❗	🛑	onlyOwner
L	addAddressesToWhitelist	Public ❗	🛑	onlyOwner
L	removeAddressFromWhitelist	Public ❗	🛑	onlyOwner
L	removeAddressesFromWhitelist	Public ❗	🛑	onlyOwner
SafeMath	**Library**			
L	add	Internal 🔒		
L	sub	Internal 🔒		
L	sub	Internal 🔒		
L	mul	Internal 🔒		
L	div	Internal 🔒		
L	div	Internal 🔒		
L	mod	Internal 🔒		
L	mod	Internal 🔒		

L	min	Internal 🔒		
L	sqrt	Internal 🔒		
GenieToken	**Implementation**	**Initializable, Context Upgradeable, ERC20 Upgradeable, UUPS Upgradeable, Whitelist**		
L	_authorizeUpgrade	Internal 🔒	🛑	onlyOwner
L	initialize	Public ❗	🛑	initializer
L	setVaultAddress	Public ❗	🛑	onlyOwner
L	setStakingAddress	Public ❗	🛑	onlyOwner
L	setMarketingAddress	Public ❗	🛑	onlyOwner
L	setPairAddress	Public ❗	🛑	onlyOwner
L	setTransactionTaxAction	Public ❗	🛑	onlyOwner
L	mint	Public ❗	🛑	onlyWhitelisted canMint
L	finishMinting	Public ❗	🛑	OnlyWhitelisted canMint
L	calculateTransactionTax	Internal 🔒		
L	transferFrom	Public ❗	🛑	NO❗
L	transfer	Public ❗	🛑	NO❗
L	calculateTransferTaxes	Public ❗		NO❗
L	remainingMintableSupply	Public ❗		NO❗

L	cap	Public ❗		NO❗
L	mintedSupply	Public ❗		NO❗
L	statsOf	Public ❗		NO❗
L	mintedBy	Public ❗		NO❗
L	setAccountCustomTax	External ❗	🛑	onlyOwner
L	removeAccountCustomTax	External ❗	🛑	onlyOwner
L	excludeAccount	External ❗	🛑	onlyOwner
L	includeAccount	External ❗	🛑	onlyOwner
L	isExcluded	Public ❗		NO❗
Initializable	**Implementation**			
L	_disableInitializers	Internal 🔒	🛑	
L	_setInitializedVersion	Private 🔐	🛑	
UUPS Upgradeable	**Implementation**	**Initializable, IERC1822Proxiable Upgradeable, ERC1967 Upgrade Upgradeable**		
L	__UUPSUpgradeable_init	Internal 🔒	🛑	onlyInitializing
L	__UUPSUpgradeable_init_unchained	Internal 🔒	🛑	onlyInitializing
L	proxiableUUID	External ❗		notDelegated
L	upgradeTo	External ❗	🛑	onlyProxy

L	upgradeToAndCall	External ❗		onlyProxy
L	_authorizeUpgrade	Internal 🔒	🛑	
Ownable Upgradeable	**Implementation**	**Initializable, Context Upgradeable**		
L	__Ownable_init	Internal 🔒	🛑	onlyInitializing
L	__Ownable_init_unchained	Internal 🔒	🛑	onlyInitializing
L	owner	Public ❗		NO❗
L	renounceOwnership	Public ❗	🛑	onlyOwner
L	transferOwnership	Public ❗	🛑	onlyOwner
L	_transferOwnership	Internal 🔒	🛑	
ERC20 Upgradeable	**Implementation**	**Initializable, Context Upgradeable, IERC20 Upgradeable, IERC20Metadata Upgradeable**		
L	__ERC20_init	Internal 🔒	🛑	onlyInitializing
L	__ERC20_init_unchained	Internal 🔒	🛑	onlyInitializing
L	name	Public ❗		NO❗
L	symbol	Public ❗		NO❗
L	decimals	Public ❗		NO❗
L	totalSupply	Public ❗		NO❗

L	balanceOf	Public ❗		NO❗
L	transfer	Public ❗	🛑	NO❗
L	allowance	Public ❗		NO❗
L	approve	Public ❗	🛑	NO❗
L	transferFrom	Public ❗	🛑	NO❗
L	increaseAllowance	Public ❗	🛑	NO❗
L	decreaseAllowance	Public ❗	🛑	NO❗
L	_transfer	Internal 🔒	🛑	
L	_mint	Internal 🔒	🛑	
L	_burn	Internal 🔒	🛑	
L	_approve	Internal 🔒	🛑	
L	_spendAllowance	Internal 🔒	🛑	
L	_beforeTokenTransfer	Internal 🔒	🛑	
L	_afterTokenTransfer	Internal 🔒	🛑	
Address Upgradeable	**Library**			
L	isContract	Internal 🔒		
L	sendValue	Internal 🔒	🛑	
L	functionCall	Internal 🔒	🛑	
L	functionCall	Internal 🔒	🛑	
L	functionCallWithValue	Internal 🔒	🛑	

L	functionCallWithValue	Internal 🔒	🛑	
L	functionStaticCall	Internal 🔒		
L	functionStaticCall	Internal 🔒		
L	verifyCallResult	Internal 🔒		
IERC1822 Proxiable Upgradeable	**Interface**			
L	proxiableUUID	External ❗		NO❗
ERC1967 Upgrade Upgradeable	**Implementation**	**Initializable**		
L	__ERC1967Upgrade_init	Internal 🔒	🛑	onlyInitializing
L	__ERC1967Upgrade_init_unchained	Internal 🔒	🛑	onlyInitializing
L	_getImplementation	Internal 🔒		
L	_setImplementation	Private 🔐	🛑	
L	_upgradeTo	Internal 🔒	🛑	
L	_upgradeToAndCall	Internal 🔒	🛑	
L	_upgradeToAndCallUUPS	Internal 🔒	🛑	
L	_getAdmin	Internal 🔒		
L	_setAdmin	Private 🔐	🛑	
L	_changeAdmin	Internal 🔒	🛑	
L	_getBeacon	Internal 🔒		

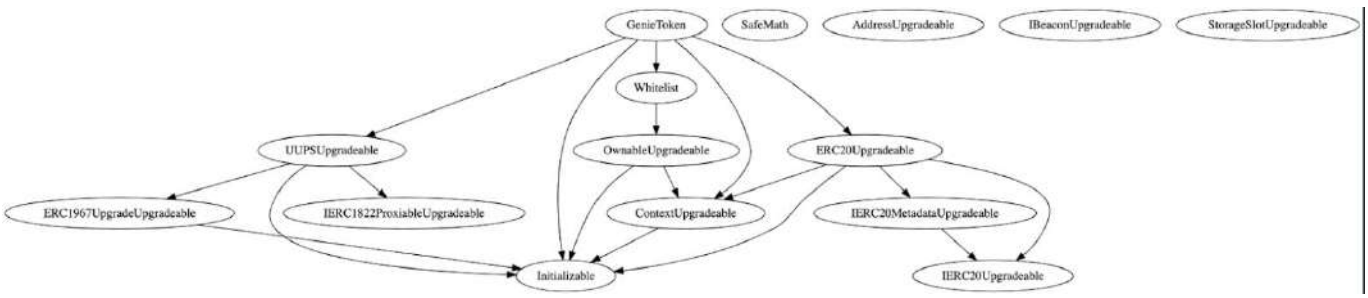
L	_setBeacon	Private 🔐	🛑	
L	_upgradeBeaconToAndCall	Internal 🔒	🛑	
L	_functionDelegateCall	Private 🔐	🛑	
IBeacon Upgradeable	**Interface**			
L	implementation	External ❗		NO❗
StorageSlot Upgradeable	**Library**			
L	getAddressSlot	Internal 🔒		
L	getBooleanSlot	Internal 🔒		
L	getBytes32Slot	Internal 🔒		
L	getUint256Slot	Internal 🔒		
Context Upgradeable	**Implementation**	**Initializable**		
L	__Context_init	Internal 🔒	🛑	onlyInitializing
L	__Context_init_unchained	Internal 🔒	🛑	onlyInitializing
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		
IERC20 Upgradeable	**Interface**			
L	totalSupply	External ❗		NO❗
L	balanceOf	External ❗		NO❗

L	transfer	External ❗	🛑	NO❗
L	allowance	External ❗		NO❗
L	approve	External ❗	🛑	NO❗
L	transferFrom	External ❗	🛑	NO❗
IERC20Metadata Upgradeable	**Interface**	**IERC20 Upgradeable**		
L	name	External ❗		NO❗
L	symbol	External ❗		NO❗
L	decimals	External ❗		NO❗

Legend

Symbol	Meaning
🛑	**Function can modify state**
💵	**Function is payable**

Inheritance Hierarchy



Security issue checking status

- **High severity issues**

No high severity issues found

- **Medium severity issues**

No medium severity issues found

- **Low severity issues**

No low severity issues found

- **Centralization issues**

❖ Whitelisted users can mint any number of tokens up to target supply

```
ftrace | funcSig
function mint(address _to, uint256 _amount) public onlyWhitelisted canMint returns(bool){
    //Never fail, just don't mint if over
    if (_amount == 0 || mintedSupply_.add(_amount) > targetSupply) {
        return false;
    }

    _mint(_to, _amount);

    mintedSupply_ = mintedSupply_.add(_amount);

    if (mintedSupply_ == targetSupply) {
        mintingFinished = true;
        emit MintFinished();
    }

    /* Members */
    if (stats[_to].txs == 0) {
        players += 1;
    }

    stats[_to].txs += 1;
    stats[_to].minted += _amount;

    totalTxs += 1;

    return true;
}
```

Owner privileges

- The owner can add/remove whitelisted users

```
 */
ftrace | funcSig
function addAddressesToWhitelist(address[] memory addrs) onlyOwner public returns(bool success) {
    for (uint256 i = 0; i < addrs.length; i++) {
        if (addAddressToWhitelist(addrs[i])) {
            success = true;
        }
    }
}

/**
 * @dev remove an address from the whitelist
 * @param addr address
 * @dev return true if the address was removed from the whitelist,
 * false if the address wasn't in the whitelist in the first place
 */
ftrace | funcSig
function removeAddressFromWhitelist(address addr) onlyOwner public returns(bool success) {
    if (whitelist[addr]) {
        whitelist[addr] = false;
        emit WhitelistedAddressRemoved(addr);
        success = true;
    }
}
```

- The owner can change staking, marketing address

```
ftrace | funcSig
function setStakingAddress(address _newStakingAddress) public onlyOwner {
    stakingAddress = _newStakingAddress;
}

ftrace | funcSig
function setMarketingAddress(address _newMarketingAddress) public onlyOwner {
    marketingAddress = _newMarketingAddress;
}
```

- The owner can change pair address

```
ftrace | funcSig
function setPairAddress(address _newPairAddress) public onlyOwner {
    pairAddress = _newPairAddress;
}
```

❖ The owner can enable/disable taxes

```
ftrace | funcSig
function setTransactionTaxAction(bool _value) public onlyOwner {
    isTaxEnable = _value;
}
```

❖ The owner and whitelisted users can mint tokens up to target supply

```
ftrace | funcSig
function mint(address _to, uint256 _amount) public onlyWhitelisted canMint returns(bool){
    //Never fail, just don't mint if over
    if (_amount == 0 || mintedSupply_.add(_amount) > targetSupply) {
        return false;
    }

    _mint(_to, _amount);

    mintedSupply_ = mintedSupply_.add(_amount);

    if (mintedSupply_ == targetSupply) {
        mintingFinished = true;
        emit MintFinished();
    }

    /* Members */
    if (stats[_to].txs == 0) {
        players += 1;
    }

    stats[_to].txs += 1;
    stats[_to].minted += _amount;

    totalTxs += 1;

    return true;
}
```

❖ The owner and whitelisted users can finish minting

```
ftrace | funcSig
function finishMinting() onlyWhitelisted canMint public returns (bool) {
    mintingFinished = true;
    emit MintFinished();
    return true;
}
```

❖ The owner can add/remove custom taxes to wallets

```
ftrace | funcSig
function setAccountCustomTax(address account, uint8 taxRate) external onlyOwner() {
    require(taxRate >= 0 && taxRate <= 100, "Invalid tax amount");
    _hasCustomTax[account] = true;
    _customTaxRate[account] = taxRate;
}

ftrace | funcSig
function removeAccountCustomTax(address account) external onlyOwner() {
    _hasCustomTax[account] = false;
}
```

❖ The owner can include/exclude wallets from fees

```
ftrace | funcSig
function excludeAccount(address account) external onlyOwner() {
    require(!_isExcluded[account], "Account is already excluded");
    _isExcluded[account] = true;
    _excluded.push(account);
}

ftrace | funcSig
function includeAccount(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _isExcluded[account] = false;
            delete _excluded[_excluded.length - 1];
            break;
        }
    }
}
```

Audit conclusion

RugFreeCoins team has performed in-depth testings, line-by-line manual code review, and automated audit of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, manipulations, and hacks. According to the smart contract audit.

Smart contract functional Status: **PASSED**

Number of risk issues: **1**

Solidity code functional issue level: **PASSED**

Number of owner privileges: **8**

Centralization risk correlated to the active owner: **HIGH**

Smart contract active ownership: **YES**

Special Note: This is a proxy contract and the owner can update the current contract and replace it with a new updated contract anytime.