Cyberscope

Audit Report

Star Link Satellite

August 2022

SHA256	2da7e7d8393fa3dba193908222488775e4f06fee2dc6358f5f36956ffdd4984c
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	StarLinkSatellite
Testing Deploy	https://testnet.bscscan.com/address/0x07Ea728B5e7e8fCb2b6565E42aEe54034E9DeB9A
Decimals	18
Domain	https://gsls-finance.web.app

Source Files

Filename	**SHA256**
contract.sol	2da7e7d8393fa3dba193908222488775e4f06fee2dc6358f5f36956ffdd4984c

Audit Updates

Initial Audit	2nd July 2022 https://github.com/cyberscope-io/audits/blob/main/gsls/v1/audit.pdf
Corrected	19th August 2022

Notes

The Star Link Satellite contract implements a token functionality enriched with a funds distribution mechanism. The functionality of the contract heavily depends on external sources. The contracts that depend on external sources should be extra careful since they could be manipulated. This audit focuses on the Star Link Satellite contract. The auditing of the external sources are out of the scope of the audit.

Contract Analysis

🔴 Critical 🟠 Medium ⚪ Minor 🔵 Pass

Severity	Code	Description
🔴	ST	Contract Owner is not able to stop or pause transactions
🔵	OCTD	Contract Owner is not able to transfer tokens from specific address
🔵	OTUT	Owner Transfer User's Tokens
🔵	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
⚪	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
🔵	MT	Contract Owner is not able to mint new tokens
🔵	BT	Contract Owner is not able to burn tokens from specific wallet
🔵	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L1938

Description

The contract owner has the authority to stop the sales for all users excluding the owner. The owner may take advantage of it by setting the `feeOnSell` to 100%. As a result, all the amount will be transferred to the contract's address.

```
function distributeTokenSellFee(address from, uint256 amountToSell) internal
returns (uint256) {
        uint256 sellFeeAmount = amountToSell * feeOnSell / 10000;
        super._transfer(from, address(this), sellFeeAmount);
```

The contract owner has the authority to stop the trades for all users including the owner. The owner may take advantage of it by setting the `swappingOnlyFromContract` to true and adding the excluded addresses to `flagOnSwap`.

```
if (swappingOnlyFromContract) {
            if (automatedMarketMakerPairs[from]) {
                require(flagOnSwap[recipient], "You are not allowed to SWAP
directly on Pancake");
            }
            if (automatedMarketMakerPairs[recipient]) {
                require(flagOnSwap[from], "You are not allowed to SWAP directly
on Pancake");
            }
}
```

Additionally, the contract owner can abuse the tokenTransferFeeDistributor address in order to produce unexpected values. As a result, the distributeTokenTransferFee can manipulate the transactions.

```
if (!swapping) {
            if (from != address(0) && (recipient == uniswapV2Pair || recipient
== priceStabilizer) && !feeExempt[from]) { // sell token
                swapping = true;
                amount -= distributeTokenSellFee(from, amount);
                swapping = false;
```

In the same manner, the contract owner can abuse the priceStabilizer address in order to produce unexpected values.

```
if (priceStabilizingEnabled) {
            if (from != address(0) && recipient == uniswapV2Pair) { // sell
token
                if (!swapping) {
                    swapping = true;
                    require(priceStabilizer != address(0), "null price
stabilizer");
                    amount =
IPriceRegulator(priceStabilizer).regulateSell(amount);
                    swapping = false;
                }
            }
```

Recommendation

The contract could embody a check for not allowing setting the feeOnSell than a reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

Regarding the external addresses read more in the corresponding section.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ULTW - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L2159

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the `distributeNodeCreationFees` method.

```
function distributeNodeCreationFees(uint256 amount) external onlyGovernor {
  _distributeNodeCreationFees(amount);
}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile the token's price.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

🔴 Critical 🟠 Medium ● Minor

Severity	Code	Description
🔴	US	Untrusted Source
●	L01	Public Function could be Declared External
●	L04	Conformance to Solidity Naming Conventions
●	L05	Unused State Variable
●	L09	Dead Code Elimination

US - Untrusted Source

Criticality	critical
Location	contract.sol#L1807,2282,1803,1996,1983

Description

The contract uses an external contract in order to determine the transaction's flow. The external contract is untrusted. As a result it may produce security issues and harm the transactions.

Many features of the contract are working as a delegator to other contracts. For instance, the auto generated liquidity pool and the swap features are delegated to a contract called "priceStabilizer". The secured functionality of these contracts is essential for the Deck contract.

The auditing of these contracts are out of the audit scope.

```
address public busdDistributor;
IFeeDistributor _feeDistributor = IFeeDistributor(tokenTransferFeeDistributor);
//
address public priceStabilizer;
amount = IPriceRegulator(priceStabilizer).regulateSell(from, amount);
```

Recommendation

- The contract should use a trusted external source. A trusted source could be either a commonly recognized or an audited contract.
- The pointing addresses should not be able to change after the initialization.
- The external calls could be wrapped by try-catch statements
- The returned value could be sanitised so the could not harm the contract’s flow.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L740,1022,1030,1118,1141,1096,1161,732,1047,1073

Description

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

```
transfer
decimals
renounceOwnership
decreaseAllowance
approve
increaseAllowance
transferFrom
symbol
name
...
```

Recommendation

Use the external attribute for functions never called from the contract.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L1498,1391,677,1371,1709,1367,702,1726,760,858,1708,1014,1363,1010,662,1395,1349,992,994,706,1541,790,659,1529,794

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.

```
__Pausable_init_unchained
_set
__Context_init
__Pausable_init
WETH
__Ownable_init_unchained
_allowances
_balances
__gap
...
```

Recommendation

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

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L760,858,1375,1793

Description

There are segments that contain unused state variables.

```
blacklistedAccounts
_NOT_ENTERED
__gap
```

Recommendation

Remove unused state variables.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L418,389,662,1318,399,364,451,702,1240,478,706,461,624,659,432

Description

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

```
functionCallWithValue
__Context_init
_disableInitializers
functionStaticCall
__Ownable_init_unchained
verifyCallResult
_burn
__Ownable_init
sendValue
...
```

Recommendation

Remove unused functions.

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		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
AddressUpgradeable	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	

	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	verifyCallResult	Internal		
Initializable	Implementation			
	_disableInitializers	Internal	✓	
	_setInitializedVersion	Private	✓	
ContextUpgradeable	Implementation	Initializable		
	__Context_init	Internal	✓	onlyInitializing
	__Context_init_unchained	Internal	✓	onlyInitializing
	_msgSender	Internal		
	_msgData	Internal		
OwnableUpgradeable	Implementation	Initializable, ContextUpgradeable		
	__Ownable_init	Internal	✓	onlyInitializing
	__Ownable_init_unchained	Internal	✓	onlyInitializing
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
PausableUpgradeable	Implementation	Initializable, ContextUpgradeable		
	__Pausable_init	Internal	✓	onlyInitializing
	__Pausable_init_unchained	Internal	✓	onlyInitializing
	paused	Public		-
	_pause	Internal	✓	whenNotPaused
	_unpause	Internal	✓	whenPaused

IERC20Upgradeable	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20MetadataUpgradeable	Interface	IERC20Upgradeable		
	name	External		-
	symbol	External		-
	decimals	External		-
ERC20Upgradeable	Implementation	Initializable, ContextUpgradeable, IERC20Upgradeable, IERC20MetadataUpgradeable		
	__ERC20_init	Internal	✓	onlyInitializing
	__ERC20_init_unchained	Internal	✓	onlyInitializing
	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	✓	
GovernableUpgradeable	Implementation	Initializable, PausableUpgradeable		
	Governable_init	External	✓	initializer
	__Governable_init	Internal	✓	onlyInitializing
	governor	Public		-
	_governor	Internal		
	_pendingGovernor	Internal		
	isGovernor	Public		-
	_setGovernor	Internal	✓	
	_setPendingGovernor	Internal	✓	
	transferGovernance	External	✓	onlyGovernor
	claimGovernance	External	✓	-
	_changeGovernor	Internal	✓	
	pause	External	✓	onlyGovernor
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-

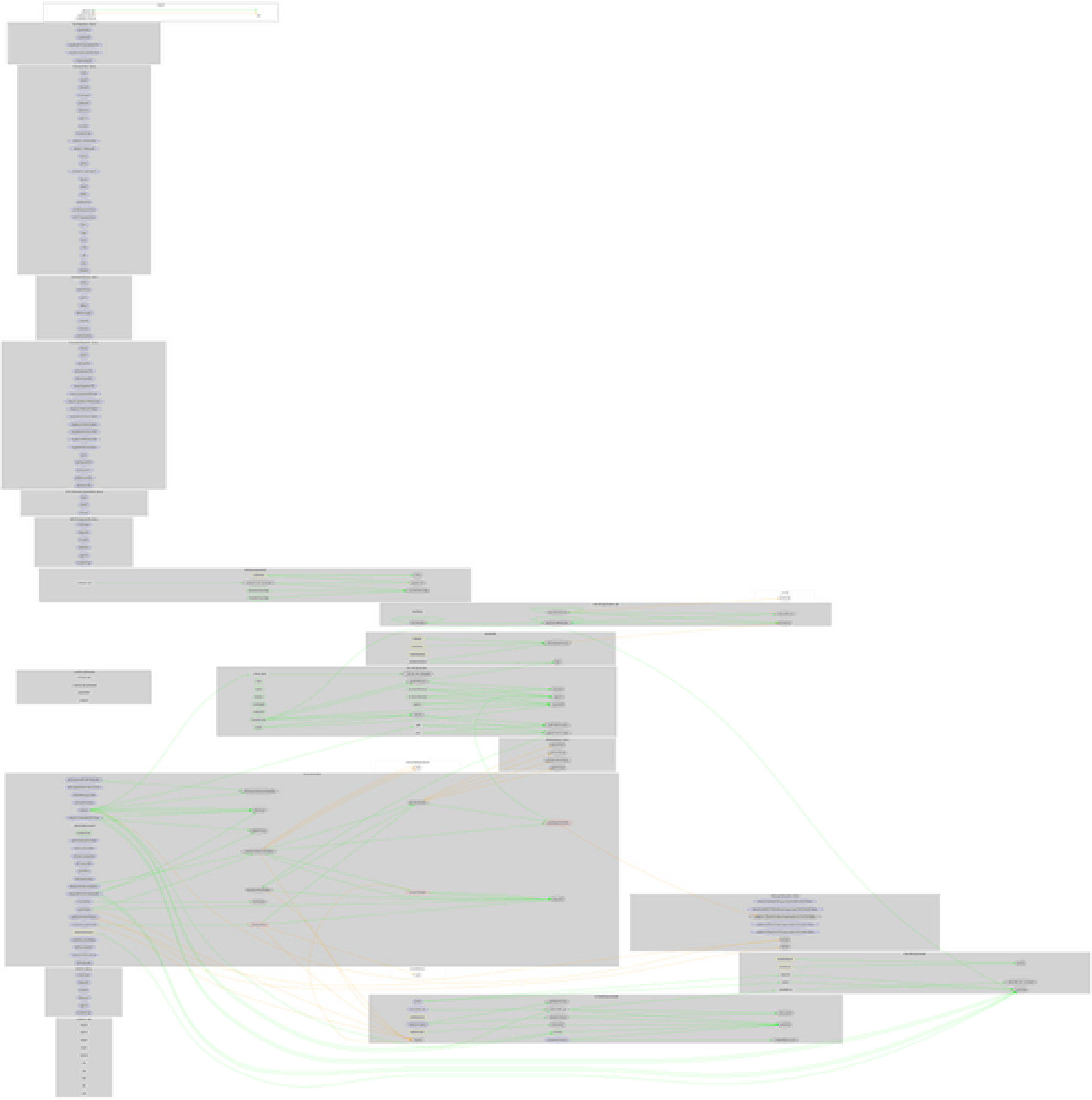
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-

	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IPriceRegulator	Interface			
	regulateBuy	External	✓	-
	regulateSell	External	✓	-
	swapBusdForStarLinkSatellite	External	✓	-
	swapStarLinkSatelliteForBusd	External	✓	-
	swapAndLiquify	External	✓	-
IFeeDistributor	Interface			
	getMaxFeeResolution	External	✓	-

	getFeeCount	External	✓	-
	getFeeShare	External	✓	-
	getFeeAddress	External	✓	-
StarLinkSatellite	Implementation	ERC20Upgradeable, GovernableUpgradeable		
	setSwappingOnlyFromContract	External	✓	onlyGovernor
	allowSwap	Internal	✓	
	disallowSwap	Internal	✓	
	setNodeManagerAddr	External	✓	onlyGovernor
	setSwapPairToken	External	✓	onlyGovernor
	initialize	External	✓	initializer
	updateUniswapV2Router	External	✓	onlyGovernor
	_beforeTokenTransfer	Internal	✓	
	transferFrom	Public	✓	-
	setDistributionThreshold	External	✓	onlyGovernor
	setRewardPoolAddr	External	✓	onlyGovernor
	setNodeCreationFees	External	✓	onlyGovernor
	setCashoutFee	External	✓	onlyGovernor
	setSellFee	External	✓	onlyGovernor
	setEnableTrading	External	✓	onlyGovernor
	setAutomatedMarketMakerPair	External	✓	onlyGovernor
	_setAutomatedMarketMakerPair	Private	✓	
	_swapAndLiquify	Private	✓	
	swapBusdForStarLinkSatellite	External	✓	-
	swapStarLinkSatelliteForBusd	External	✓	-
	_swapTokensForBUSD	Private	✓	
	payForNode	External	✓	onlyNodeManager
	distributeNodeCreationFees	External	✓	onlyGovernor
	_distributeNodeCreationFees	Private	✓	
	_feeOnCashout	Private	✓	

	cashoutRewardToNoder	External	✓	onlyNodeManager
	setAntiWhale	External	✓	onlyGovernor
	enablePriceStabilizing	External	✓	onlyGovernor
	setPriceStabilizer	External	✓	onlyGovernor
	updateBUSDDistributor	External	✓	onlyGovernor
	setFeeExempt	External	✓	onlyGovernor
	distributeTokenSellFee	Internal	✓	
	distributeBUSD	Internal	✓	

Contract Flow



Domain Info

Domain Name	web.app
Registry Domain ID	300A2C851-APP
Creation Date	2019-01-08T22:05:04Z
Updated Date	2021-12-12T09:32:53Z
Registry Expiry Date	2023-01-08T22:05:04Z
Registrar WHOIS Server	whois.nic.google
Registrar URL	http://www.markmonitor.com
Registrar	MarkMonitor Inc.
Registrar IANA ID	292

The domain has been created in 5 months before the creation of the audit.

There is no public billing information, the creator is protected by the privacy settings.

Summary

There are some functions that can be abused by the owner like stopping transactions and transferring funds to the team’s wallet. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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