



**SOLIDProof**  
*Bring trust into your projects*

**Blockchain Security | Smart Contract Audits | KYC**

MADE IN GERMANY

# **Awake Audit**

**Security Assessment  
26. April, 2022**

**For**



Disclaimer	3
Description	5
Project Engagement	5
Logo	5
Contract Link	5
Methodology	7
Used Code from other Frameworks/Smart Contracts (direct imports)	8
Tested Contract Files	9
Source Lines	10
Risk Level	10
Capabilities	11
Inheritance Graph	12
CallGraph	13
Scope of Work/Verify Claims	14
Modifiers and public functions	20
Source Units in Scope	23
Critical issues	24
High issues	24
Medium issues	24
Low issues	24
Informational issues	25
Audit Comments	25
SWC Attacks	26

## Disclaimer

SolidProof.io reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc’...)

**SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.**

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof’s position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	26. April 2022	<ul style="list-style-type: none"><li>• Layout project</li><li>• Automated- /Manual-Security Testing</li><li>• Summary</li></ul>

## **Network**

Binance Smart Chain (BEP20)

## **Website**

<https://awakeinthegalactic.com/>

## **Telegram**

<https://t.me/awakeinthegalactic1>

## **Twitter**

[https://twitter.com/galactic\\_awake](https://twitter.com/galactic_awake)

## **Instagram**

<https://www.instagram.com/awakeinthegalactic/>

## **Facebook**

<https://www.instagram.com/awakeinthegalactic/>

## **Discord**

<https://discord.gg/wJ4D5JSY>

## **Youtube**

[https://www.youtube.com/channel/UCe1\\_7\\_XmO1km8pD\\_3yUOK\\_Q](https://www.youtube.com/channel/UCe1_7_XmO1km8pD_3yUOK_Q)

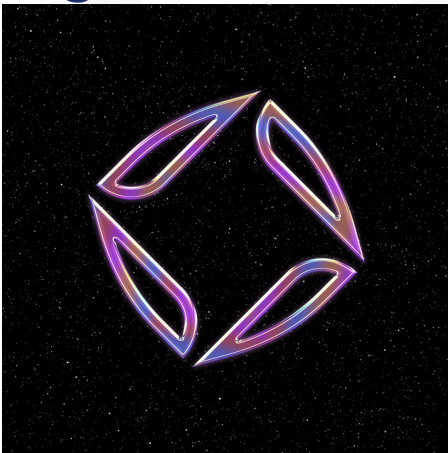
## Description

AWAKE is built on a strong and loyal community base gained through high quality experiences. Our community will be able to enjoy the following activities on the platform.

## Project Engagement

During the 25th of April 2022, **AWAKE Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

## Logo



## Contract Link

**v1.0**

- <https://bscscan.com/address/0x173496f1473Db153e4bc98D50546D46184064D8C>

# Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
<b>Critical</b>	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
<b>High</b>	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
<b>Medium</b>	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
<b>Low</b>	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
<b>Informational</b>	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

# Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

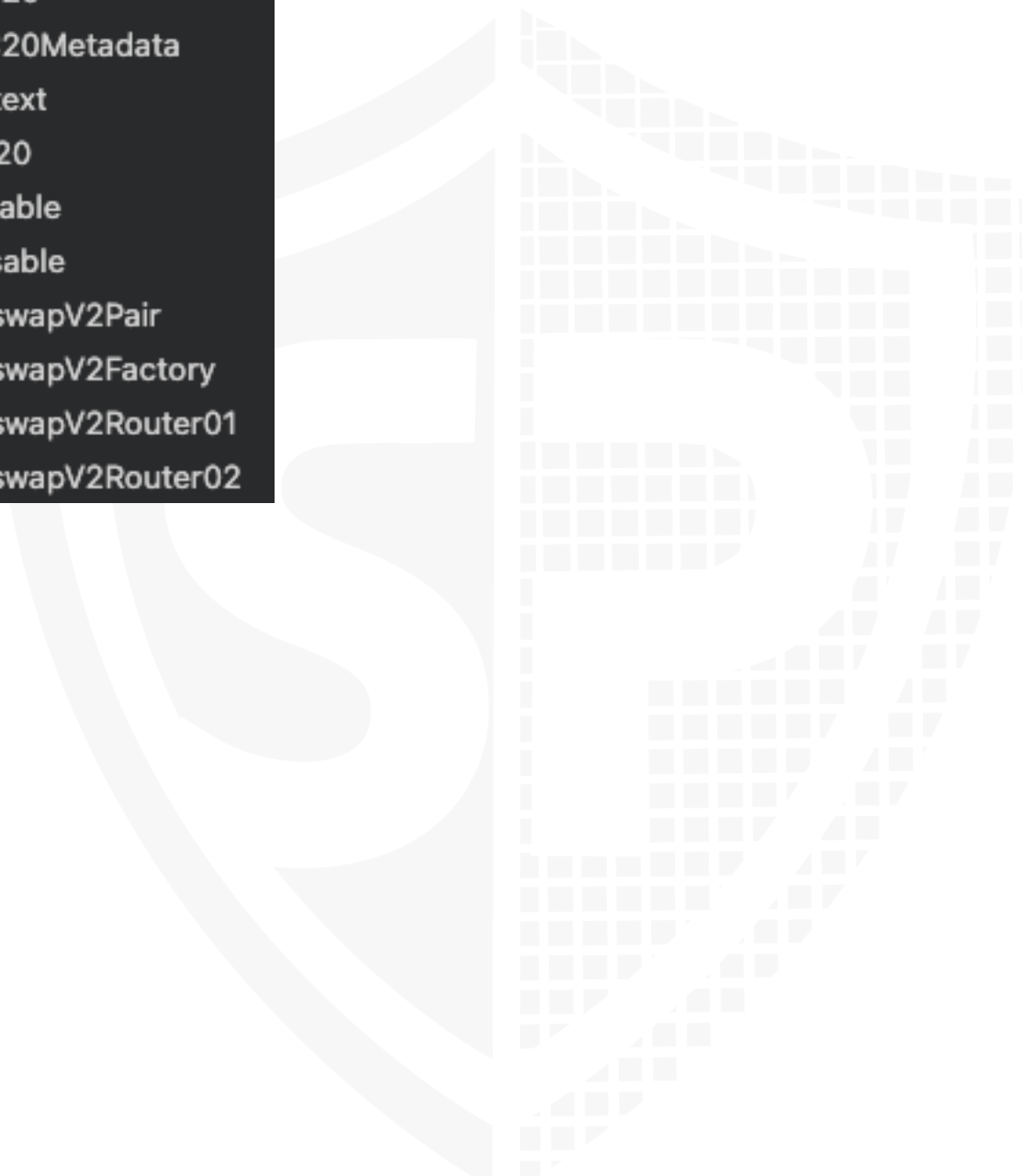
## **Methodology**

The auditing process follows a routine series of steps:

1. Code review that includes the following:
  - i) Review of the specifications, sources, and instructions provided to SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
  - ii) Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
  - iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SolidProof describe.
2. Testing and automated analysis that includes the following:
  - i) Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
  - ii) Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

## Used Code from other Frameworks/Smart Contracts (direct imports)

Imported packages:



```
IERC20
IERC20Metadata
Context
ERC20
Ownable
Pausable
IUniswapV2Pair
IUniswapV2Factory
IUniswapV2Router01
IUniswapV2Router02
```



## Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

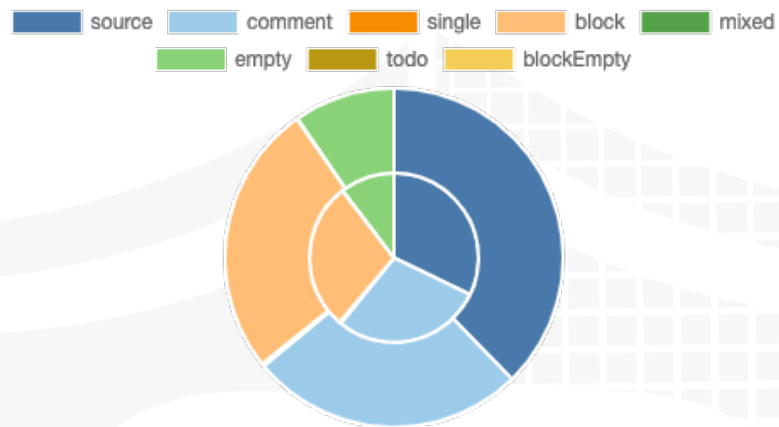
*A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.*

### v1.0

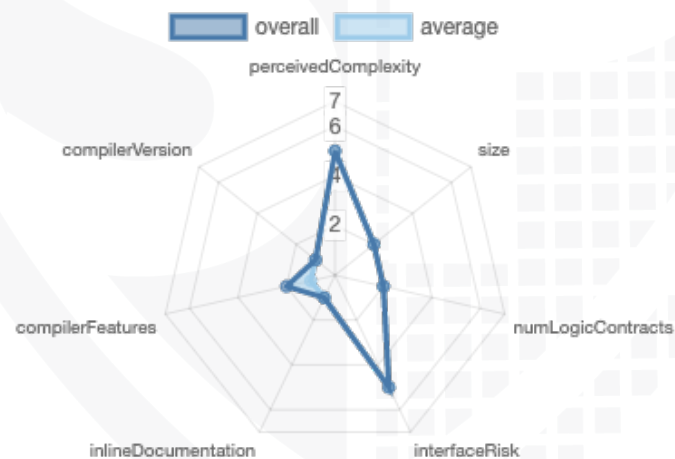
File Name	SHA-1 Hash
contracts/awake.sol	6ab4e43fac4ce4cf2cfc5952ba25ff73d0cebe00

# Metrics

## Source Lines v1.0



## Risk Level v1.0



## Capabilities

### Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	2	0	6	3

### Exposed Functions

*This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.*

Version	Public	Payable
1.0	99	6

Version	External	Internal	Private	Pure	View
1.0	69	75	1	10	37

### State Variables

Version	Total	Public
1.0	35	1

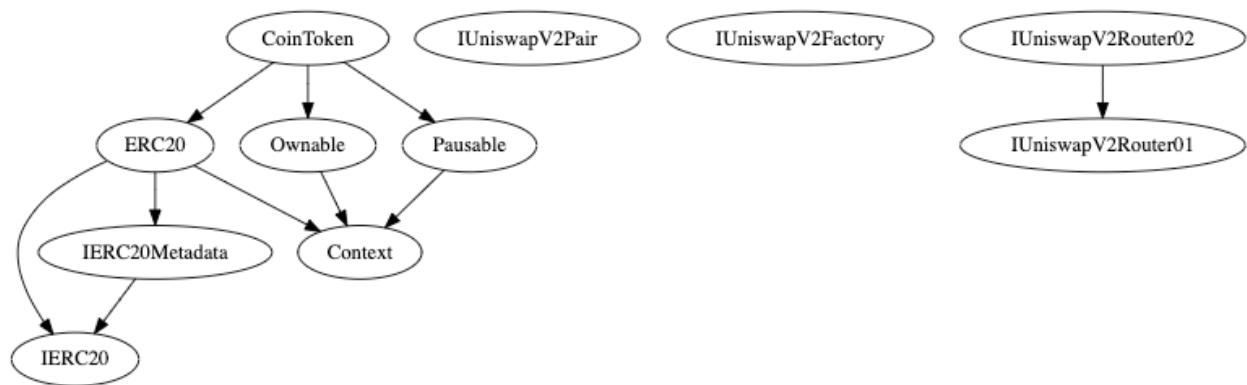
### Capabilities

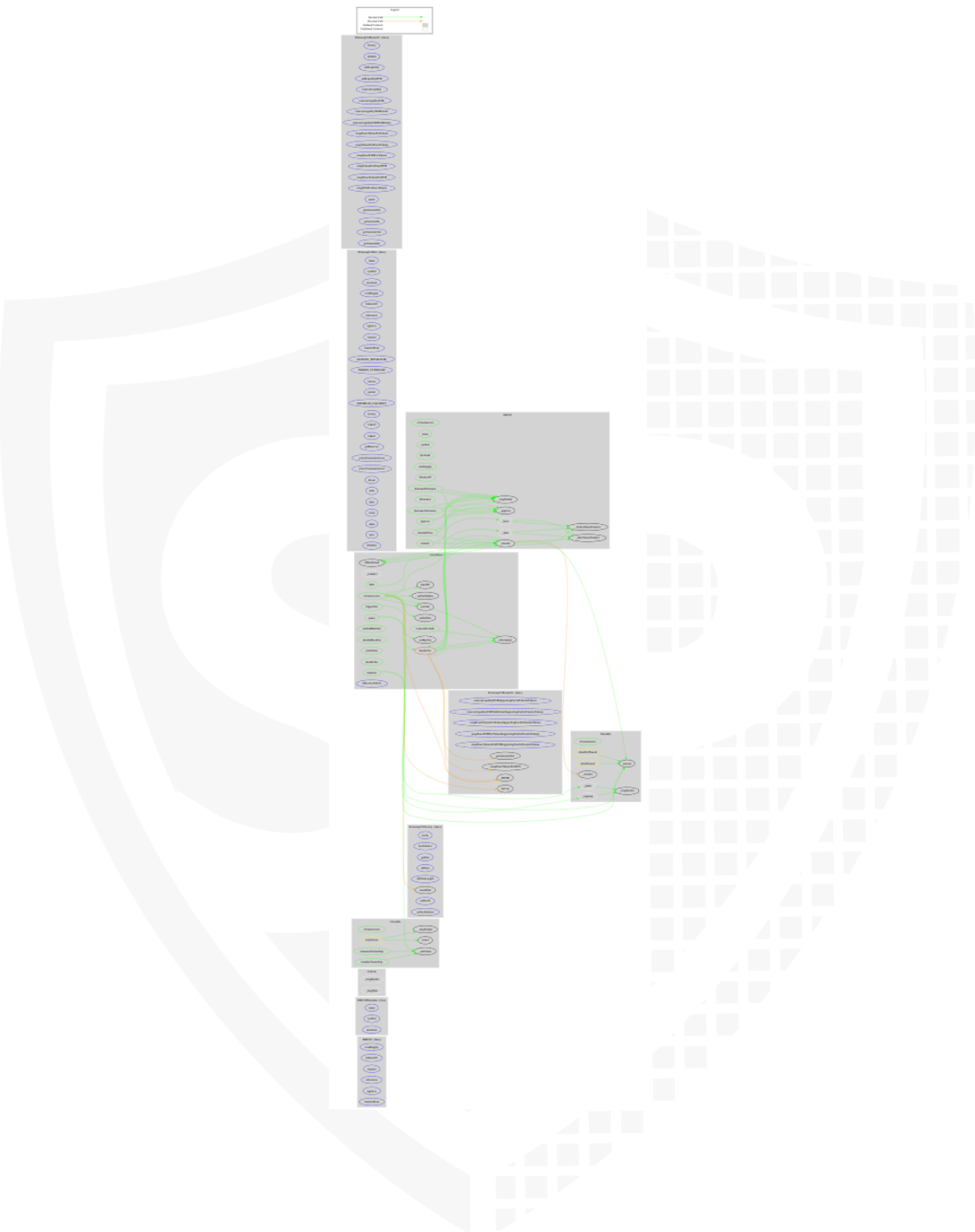
Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	^0.8.9		yes	yes (3 asm blocks)	

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	EC Recover	New/Create/Create2
---------	---------------	-----------------	--------------	---------------------	------------	--------------------

1.0	yes					
-----	-----	--	--	--	--	--

## Inheritance Graph v1.0





## Scope of Work/Verify Claims

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

1. Correct implementation of Token standard
2. Deployer cannot mint any new tokens
3. Deployer cannot burn or lock user funds
4. Deployer cannot pause the contract
5. Overall checkup (Smart Contract Security)

### Correct implementation of Token standard

ERC20				
Function	Description	Exist	Tested	Verified
TotalSupply	Provides information about the total token supply	✓	✓	✓
BalanceOf	Provides account balance of the owner's account	✓	✓	✓
Transfer	Executes transfers of a specified number of tokens to a specified address	✓	✓	✓
TransferFrom	Executes transfers of a specified number of tokens from a specified address	✓	✓	✓
Approve	Allow a spender to withdraw a set number of tokens from a specified account	✓	✓	✓
Allowance	Returns a set number of tokens from a spender to the owner	✓	✓	✓

## Write functions of contract v1.0

1. approve

2. burn

3. decreaseAllowance

4. disableBlacklist

5. disableTax

6. enableBlacklist

7. enableTax

8. exclude

9. increaseAllowance

10. pause

11. removeExclude

12. renounceOwnership

13. setBuyTax

14. setSellTax

15. setTaxWallets

16. transfer

17. transferFrom

18. transferOwnership

19. triggerTax

20. unpause

## Deployer cannot mint any new tokens

Name	Exist	Tested	Status
Deployer cannot mint	✓	✓	✓
Max / Total Supply	10.000.000		





## Deployer cannot burn or lock user funds

Name	Exist	Tested	Status
Deployer cannot lock	✓	✓	✗
Deployer cannot burn	✓	✓	✗

Comments:

### v1.0

- Owner can burn own tokens
- Owner can blacklist addresses

## Deployer cannot pause the contract

Name	Exist	Tested	Status
Deployer cannot pause	✓	✓	✗

Comments:

**v1.0**

- Owner can pause contract



## Overall checkup (Smart Contract Security)

Tested	Verified
✓	✓

### Legend

Attribute	Symbol
Verified / Checked	✓
Partly Verified	⚠
Unverified / Not checked	✗
Not available	—

# Modifiers and public functions

v1.0

▼ 🔹 <b>triggerTax</b>	🔒 onlyOwner
▼ 🔹 <b>pause</b>	🔒 onlyOwner
▼ 🔹 <b>unpause</b>	🔒 onlyOwner
▼ 🔹 <b>burn</b>	🔒 onlyOwner
▼ 🔹 <b>enableBlacklist</b>	🔒 onlyOwner
▼ 🔹 <b>disableBlacklist</b>	🔒 onlyOwner
▼ 🔹 <b>exclude</b>	🔒 onlyOwner
▼ 🔹 <b>removeExclude</b>	🔒 onlyOwner
▼ 🔹 <b>setBuyTax</b>	🔒 onlyOwner
▼ 🔹 <b>setSellTax</b>	🔒 onlyOwner
▼ 🔹 <b>setTaxWallets</b>	🔒 onlyOwner
▼ 🔹 <b>enableTax</b>	🔒 onlyOwner
▼ 🔹 <b>disableTax</b>	🔒 onlyOwner
▼ 🔹 <b>renounceOwnership</b>	🔒 onlyOwner
▼ 🔹 <b>transferOwnership</b>	🔒 onlyOwner

## Comments

- Deployer can set following state variables without any limitations
  - sellTaxes["dev"]
  - sellTaxes["marketing"]
  - sellTaxes["liquidity"]
  - sellTaxes["charity"]
  - buyTaxes["dev"]
  - buyTaxes["marketing"]
  - buyTaxes["liquidity"]
  - buyTaxes["charity"]





- Deployer can enable/disable following state variables
  - taxStatus
  - excludeList
  - blacklist
  - \_paused
- Deployer can set following addresses
  - taxWallets["dev"]
  - taxWallets["marketing"]
  - taxWallets["charity"]
- We recommend you to move L895-L897 into the "else if" condition (L916) because that's the only place where it is used
- If an address send an amount to another address which is not from/to uniswap router address there are no taxes for it (for more information look at handleTax function in L894)
- Pseudocode test with calling more times the \_transfer function while handle tax. It is a recursive call of the handleTax function:
  - Test result for sending 1 Ether with 5% taxes
    - Amount: 10000000000000000000 Calc: 5000000000000000000
    - Amount: 5000000000000000000 Calc: 2500000000000000000
    - Amount: 2500000000000000000 Calc: 1250000000000000000
    - Amount: 1250000000000000000 Calc: 625000000000000000
    - Amount: 625000000000000000 Calc: 312500000000000000
    - Amount: 312500000000000000 Calc: 156250000000000000
    - Amount: 156250000000000000 Calc: 78125000000000000
    - Amount: 78125000000000000 Calc: 39062500000000000
    - Amount: 39062500000000000 Calc: 19531250000000000
    - Amount: 19531250000000000 Calc: 9765625000000000
    - Amount: 9765625000000000 Calc: 4882812500000000
    - Amount: 4882812500000000 Calc: 2441406250000000
    - Amount: 2441406250000000 Calc: 1220703125000000
    - Amount: 1220703125000000 Calc: 610351562500000
    - Amount: 610351562500000 Calc: 305175781250000
    - Amount: 305175781250000 Calc: 152587890625000
    - Amount: 152587890625000 Calc: 76293945312500
    - Amount: 76293945312500 Calc: 38146972656250
    - Amount: 38146972656250 Calc: 19073486328125
    - Amount: 19073486328125 Calc: 9536743164062
    - Amount: 9536743164062 Calc: 4768371582031
    - Amount: 4768371582031 Calc: 2384185791015
    - Amount: 2384185791015 Calc: 1192092895508
    - Amount: 1192092895508 Calc: 596046447754
    - Amount: 596046447754 Calc: 298023223877
    - Amount: 298023223877 Calc: 149011611938
    - Amount: 149011611938 Calc: 74505805969
    - Amount: 74505805969 Calc: 37252902984
    - Amount: 37252902984 Calc: 18626451492
    - Amount: 18626451492 Calc: 9313225746
    - Amount: 9313225746 Calc: 4656612873
    - Amount: 4656612873 Calc: 2328306436
    - Amount: 2328306436 Calc: 1164153218
    - Amount: 1164153218 Calc: 582076609
    - Amount: 582076609 Calc: 291038304
    - Amount: 291038304 Calc: 145519152
    - Amount: 145519152 Calc: 72759576
    - Amount: 72759576 Calc: 36379788
    - Amount: 36379788 Calc: 18189894
    - Amount: 18189894 Calc: 9094947
    - Amount: 9094947 Calc: 4547473
    - Amount: 4547473 Calc: 2273736
    - Amount: 2273736 Calc: 1136868
    - Amount: 1136868 Calc: 568434
    - Amount: 568434 Calc: 284217
    - Amount: 284217 Calc: 142108
    - Amount: 142108 Calc: 71054
    - Amount: 71054 Calc: 35527
    - Amount: 35527 Calc: 17763
    - Amount: 17763 Calc: 8881
    - Amount: 8881 Calc: 4440
    - Amount: 4440 Calc: 2220
    - Amount: 2220 Calc: 1110
    - Amount: 1110 Calc: 555
    - Amount: 555 Calc: 277
    - Amount: 277 Calc: 138
    - Amount: 138 Calc: 69
    - Amount: 69 Calc: 34
    - Amount: 34 Calc: 17
    - Amount: 17 Calc: 8
    - Amount: 8 Calc: 4
    - Amount: 4 Calc: 2
    - Amount: 2 Calc: 1
    - Amount: 1 Calc: 0
  - And after the handleTax is done the returning of rest amount is following
    - Amount: 12 RestAmount: 12
    - Amount: 244 RestAmount: 232
    - Amount: 4882 RestAmount: 4638
    - Amount: 97656 RestAmount: 92774
    - Amount: 1953125 RestAmount: 1855469
    - Amount: 39062500 RestAmount: 37109375
    - Amount: 781250000 RestAmount: 742187500

- Amount: 15625000000 RestAmount: 14843750000
- Amount: 312500000000 RestAmount: 296875000000
- Amount: 6250000000000 RestAmount: 5937500000000
- Amount: 125000000000000 RestAmount: 118750000000000
- Amount: 2500000000000000 RestAmount: 2375000000000000
- Amount: 50000000000000000 RestAmount: 47500000000000000
- Amount: 1000000000000000000 RestAmount: 950000000000000000
- Result balances
  - Receiver balance: BigNumber { value: "950000000000000000" }
  - Sender balance: BigNumber { value: "9999990000000000000000000000" }
- You can achieve the same result if you are calling super.\_transfer function in the handleTax function because you want to only the taxes without handling tax for the tax

**Please check if an OnlyOwner or similar restrictive modifier has been forgotten.**

# Source Units in Scope

## v1.0

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/awake.sol	5	6	1140	845	382	407	458	
	<b>Totals</b>	<b>5</b>	<b>6</b>	<b>1140</b>	<b>845</b>	<b>382</b>	<b>407</b>	<b>458</b>	

### Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
nSLOC	normalized source lines of code (only source-code lines; no comments, no blank lines)
Comment Lines	lines containing single or block comments
Complexity Score	a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

# Audit Results

## AUDIT PASSED

### Critical issues

No critical issues

### High issues

No high issues

### Medium issues

No medium issues

### Low issues

Issue	File	Type	Line	Description
#1	Main	Contract doesn't import npm packages from source (like OpenZeppelin etc.)	-	We recommend to import all packages from npm directly without flatten the contract. Functions could be modified or can be susceptible to vulnerabilities
#2	Main	A floating pragma is set	8	The current pragma Solidity directive is „^0.8.9”.
#3	Main	Transfer in handleTax	See description	<p>We would recommend you to call super._transfer Instead of _transfer function (L909) in handleTax function (L894) because of you want to transfer only the tax without handling the taxes.</p> <p>For more information look at modifier section</p>



#4	Main	Unchecked low-level calls	978-980, 983	Ensure that the return value of a low-level call is checked or logged
----	------	---------------------------	--------------	---

## Informational issues

Issue	File	Type	Line	Description
#1	Main	State variables that could be declared constant	835, 837	Add the `constant` attributes to state variables that never change
#2	Main	Unused state variables	850, 845, 840, 851, 846, 841, 849, 844, 839, 852, 847, 842	Remove unused state variables

## Audit Comments

We recommend you to use the special form of comments (NatSpec Format, Follow link for more information <https://docs.soliditylang.org/en/v0.5.10/natspec-format.html>) for your contracts to provide rich documentation for functions, return variables and more. This helps investors to make clear what that variables, functions etc. do.

### 26. April 2022:

- Read whole report for more information

## SWC Attacks

ID	Title	Relationships	Status
<a href="#">SW C-1 36</a>	Unencrypted Private Data On-Chain	<a href="#">CWE-767: Access to Critical Private Variable via Public Method</a>	PASSED
<a href="#">SW C-1 35</a>	Code With No Effects	<a href="#">CWE-1164: Irrelevant Code</a>	PASSED
<a href="#">SW C-1 34</a>	Message call with hardcoded gas amount	<a href="#">CWE-655: Improper Initialization</a>	PASSED
<a href="#">SW C-1 33</a>	Hash Collisions With Multiple Variable Length Arguments	<a href="#">CWE-294: Authentication Bypass by Capture-replay</a>	PASSED
<a href="#">SW C-1 32</a>	Unexpected Ether balance	<a href="#">CWE-667: Improper Locking</a>	PASSED
<a href="#">SW C-1 31</a>	Presence of unused variables	<a href="#">CWE-1164: Irrelevant Code</a>	NOT PASSED
<a href="#">SW C-1 30</a>	Right-To-Left-Override control character (U+202E)	<a href="#">CWE-451: User Interface (UI) Misrepresentation of Critical Information</a>	PASSED
<a href="#">SW C-1 29</a>	Typographical Error	<a href="#">CWE-480: Use of Incorrect Operator</a>	PASSED
<a href="#">SW C-1 28</a>	DoS With Block Gas Limit	<a href="#">CWE-400: Uncontrolled Resource Consumption</a>	PASSED

<a href="#">SW C-1 27</a>	Arbitrary Jump with Function Type Variable	<a href="#">CWE-695: Use of Low-Level Functionality</a>	<b>PASSED</b>
<a href="#">SW C-1 25</a>	Incorrect Inheritance Order	<a href="#">CWE-696: Incorrect Behavior Order</a>	<b>PASSED</b>
<a href="#">SW C-1 24</a>	Write to Arbitrary Storage Location	<a href="#">CWE-123: Write-what-where Condition</a>	<b>PASSED</b>
<a href="#">SW C-1 23</a>	Requirement Violation	<a href="#">CWE-573: Improper Following of Specification by Caller</a>	<b>PASSED</b>
<a href="#">SW C-1 22</a>	Lack of Proper Signature Verification	<a href="#">CWE-345: Insufficient Verification of Data Authenticity</a>	<b>PASSED</b>
<a href="#">SW C-1 21</a>	Missing Protection against Signature Replay Attacks	<a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>	<b>PASSED</b>
<a href="#">SW C-1 20</a>	Weak Sources of Randomness from Chain Attributes	<a href="#">CWE-330: Use of Insufficiently Random Values</a>	<b>PASSED</b>
<a href="#">SW C-11 9</a>	Shadowing State Variables	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>
<a href="#">SW C-11 8</a>	Incorrect Constructor Name	<a href="#">CWE-665: Improper Initialization</a>	<b>PASSED</b>
<a href="#">SW C-11 7</a>	Signature Malleability	<a href="#">CWE-347: Improper Verification of Cryptographic Signature</a>	<b>PASSED</b>

<a href="#">SW C-11 6</a>	Timestamp Dependence	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	<b>PASSED</b>
<a href="#">SW C-11 5</a>	Authorization through tx.origin	<a href="#">CWE-477: Use of Obsolete Function</a>	<b>PASSED</b>
<a href="#">SW C-11 4</a>	Transaction Order Dependence	<a href="#">CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')</a>	<b>PASSED</b>
<a href="#">SW C-11 3</a>	DoS with Failed Call	<a href="#">CWE-703: Improper Check or Handling of Exceptional Conditions</a>	<b>PASSED</b>
<a href="#">SW C-11 2</a>	Delegatecall to Untrusted Callee	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	<b>PASSED</b>
<a href="#">SW C-11 1</a>	Use of Deprecated Solidity Functions	<a href="#">CWE-477: Use of Obsolete Function</a>	<b>PASSED</b>
<a href="#">SW C-11 0</a>	Assert Violation	<a href="#">CWE-670: Always-Incorrect Control Flow Implementation</a>	<b>PASSED</b>
<a href="#">SW C-1 09</a>	Uninitialized Storage Pointer	<a href="#">CWE-824: Access of Uninitialized Pointer</a>	<b>PASSED</b>
<a href="#">SW C-1 08</a>	State Variable Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>
<a href="#">SW C-1 07</a>	Reentrancy	<a href="#">CWE-841: Improper Enforcement of Behavioral Workflow</a>	<b>PASSED</b>
<a href="#">SW C-1 06</a>	Unprotected SELFDESTRUCT Instruction	<a href="#">CWE-284: Improper Access Control</a>	<b>PASSED</b>

<a href="#">SW</a> <a href="#">C-1</a> <a href="#">05</a>	Unprotected Ether Withdrawal	<a href="#">CWE-284: Improper Access Control</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">04</a>	Unchecked Call Return Value	<a href="#">CWE-252: Unchecked Return Value</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">03</a>	Floating Pragma	<a href="#">CWE-664: Improper Control of a Resource Through its Lifetime</a>	<b>NOT PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">02</a>	Outdated Compiler Version	<a href="#">CWE-937: Using Components with Known Vulnerabilities</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">01</a>	Integer Overflow and Underflow	<a href="#">CWE-682: Incorrect Calculation</a>	<b>PASSED</b>
<a href="#">SW</a> <a href="#">C-1</a> <a href="#">00</a>	Function Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	<b>PASSED</b>

The logo features the words "Solid Proofed" in a white, elegant script font. The word "Solid" is positioned above "Proofed". Behind the text is a faint, stylized shield emblem with a grid pattern, set against a solid blue background.

Solid  
Proofed

**Blockchain Security | Smart Contract Audits | KYC**

A horizontal bar representing the German flag, with black, red, and gold stripes.

MADE IN GERMANY