



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

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

MADE IN GERMANY

**Supontis**

**Audit**

**Security Assessment  
23. September, 2022**

**For**



**SolidProof\_io**



**@solidproof\_io**

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	13
CallGraph	14
Scope of Work/Verify Claims	15
Modifiers and public functions	25
Source Units in Scope	27
Critical issues	28
High issues	28
Medium issues	28
Low issues	28
Informational issues	28
Audit Comments	29
SWC Attacks	30

## 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	23. September 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

- Supontis
  - <https://supontis.com>

## Telegram

- Supontis
  - <https://t.me/SuontisTokenOfficial>

## Twitter

- Supontis
  - [https://twitter.com/Suontis\\_Token](https://twitter.com/Suontis_Token)



## Description

TBA

## Project Engagement

During the 20th of September 2022, **SDR 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

• TBA

# 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:

Dependency / Import Path	Count
@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol	3
@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol	3
@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol	2
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20VotesUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/draft-ERC20PermitUpgradeable.sol	1
@openzeppelin/contracts/token/ERC20/ERC20.sol	1



## Tested Contract Files

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

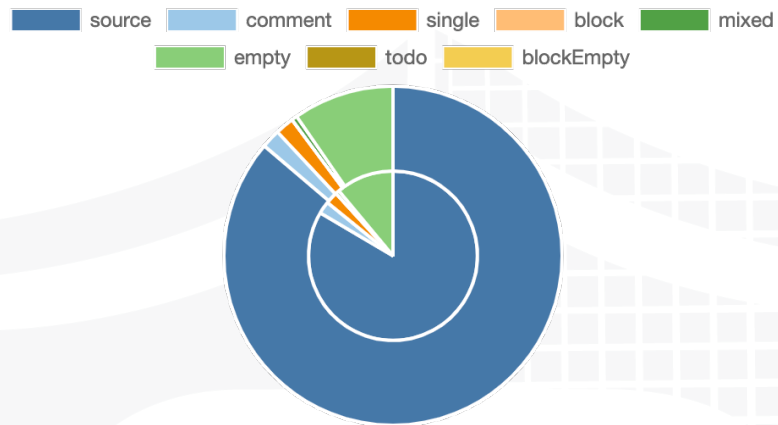
*A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.*

### v1.0

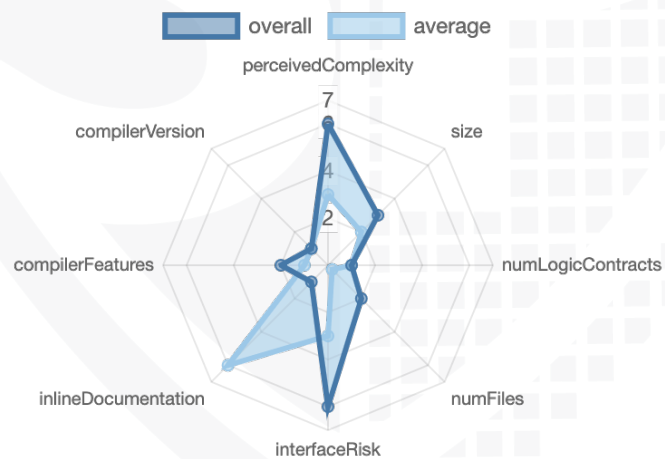
File Name	SHA-1 Hash
contracts/IPancakeRouter01.sol	9a367f28c9eebde50607e801fad647f225dcb9fe
contracts/IPancakePair.sol	116aa17ebf10b73f8622ad7c3e44d9c7cd73557f
contracts/Dogeliens.sol	a4506162332cafe63992da84ac4c62517e6a66fd
contracts/IPancakeRouter02.sol	0260bab1f74d0a24d3cbf8ad06f0627c91368ba1
contracts/PancakeCaller.sol	e63ad4788f92b3dc619790e4959783df759b558c
contracts/Supontis.sol	9fb5d27d0b71ee408860289c26f857fb745dcdd4
contracts/IPancakeFactory.sol	5e466f0f0a1467b0ed261b1a9bcc5e81cb31d405
contracts/Rocketize.sol	8c5bffe81637be920f2bb9ff39c9d43b18e3194f

# Metrics

## Source Lines v1.0



## Risk Level v1.0



## Capabilities

### Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	4	0	9	0

### 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	189	11

Version	External	Internal	Private	Pure	View
1.0	154	164	51	15	58

## State Variables

Version	Total	Public
1.0	131	75

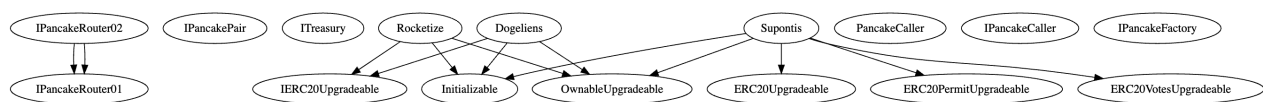
## Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	>=0.6.2 >=0.5.0 ^0.8.11 ^0.8.13 0.8.15		yes		

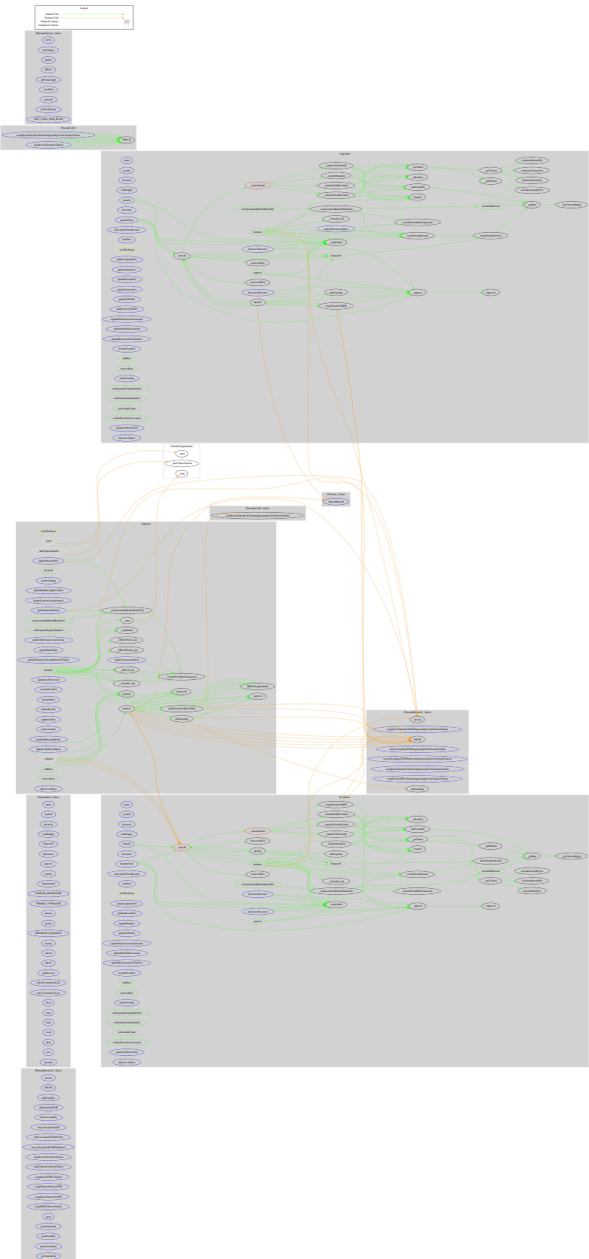
Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	EC Recover	New/Create/Create2
1.0	yes					

# Inheritance Graph

## v1.0



# CallGraph 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. Is contract an upgradeable
2. Correct implementation of Token standard
3. Deployer cannot mint any new tokens
4. Deployer cannot burn or lock user funds
5. Deployer cannot pause the contract
6. Deployer cannot set fees
7. Deployer cannot blacklist/antisnipe addresses
8. Overall checkup (Smart Contract Security)

## Is contract an upgradeable

Name	
Is contract an upgradeable?	Yes

Comments:

### v1.0

- Owner can deploy a new version of the contract which can change any limit and give owner new privileges
  - Be aware of this and do your own research for the contract which is the contract pointing to



## 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

▼ SUPONTIS

Balance: 0 ETH

addBots

approve

decreaseA...

delegate

delegateB...

enableTra...

excludeFro...

excludeFro...

includeInF...

increaseAL...

initialize

permit

removeBots

removeLi...

renounce...

setAutom...

setSwapA...

transfer

transferFr...

transferO...

updateBuy...

updateGas...

updateIsM...

updateLiq...

updateMar...

updateMa...

updateMa...

updateMin...

updatePan...

updatePan...

updateSell...

updateTra...

updateTre...

## Deployer cannot mint any new tokens

Name	Exist	Tested	Status
Deployer cannot mint	✓	✓	✓



## Deployer cannot burn or lock user funds

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

Comments:

### v1.0

- Owner can lock user funds by
  - blacklisting addresses
  - Setting max tx amount to 0
  - Setting max wallet amount to 0
    - We recommend you to set these variables above 0
- Tokens
  - will be burned while tx

## Deployer cannot pause the contract

Name	Exist	Tested	Status
Deployer cannot pause	✓	✓	✓



## Deployer cannot set fees

Name	Exist	Tested	Status
Deployer cannot set fees over 25%	✓	✓	✗
Deployer cannot set fees to nearly 100% or to 100%	✓	✓	✓

Comments:

### v1.0

- Fees can be set above 25%

## Deployer can blacklist/antisnipe addresses

Name	Exist	Tested	Status
Deployer cannot blacklist/antisnipe addresses	✓	✓	✗

Comments:

**v1.0**

- Owner can blacklisted addresses



## 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

Supontis

Initialize	
Ⓜ initializer	
enableTrading	
Ⓜ onlyOwner	
updateIsMarketingFeeToken	
Ⓜ onlyOwner	
updateTransferDelayEnabled	
Ⓜ onlyOwner	
updatePancakePair	
Ⓜ onlyOwner	
updatePancakeRouter	
Ⓜ onlyOwner	
setSwapAndLiquifyEnabled	
Ⓜ onlyOwner	
updateMaxTransactionAmount	
Ⓜ onlyOwner	
updateMaxWallet	
Ⓜ onlyOwner	
updateMinimumTokensBeforeFeeTaken	
Ⓜ onlyOwner	
setAutomatedMarketMakerPair	
Ⓜ onlyOwner	
updateGasPriceLimit	
Ⓜ onlyOwner	
excludeFromMaxTransaction	
Ⓜ onlyOwner	
excludeFromFee	
Ⓜ onlyOwner	
includeInFee	
Ⓜ onlyOwner	
updateBuyFee	
Ⓜ onlyOwner	
updateSellFee	
Ⓜ onlyOwner	
removeLimits	
Ⓜ onlyOwner	
updateTreasuryAddress	
Ⓜ onlyOwner	
updateLiquidityAddress	
Ⓜ onlyOwner	
updateMarketingWallet	
Ⓜ onlyOwner	
addBots	
Ⓜ onlyOwner	
removeBots	
Ⓜ onlyOwner	

## Comments

















- Deployer can set following state variables without any limitations
  - **Supontis**
    - maxTransactionAmount
    - maxWallet
    - minimumFeeTokensToTake
    - \_gasPriceLimit
- Deployer can enable/disable following state variables

- **Supontis**
  - isMarketingFeeBaseToken
  - transferDelayEnabled
  - swapAndLiquifyEnabled
  - automatedMarketMakerPairs
  - isExcludedMaxTransactionAmount
  - isExcludedFromFee
  - isTreasuryContract
  - bots
- Deployer can set following addresses
  - **Supontis**
    - baseTokenForPair
    - pancakePair
    - pancakeRouter
    - treasuryAddress
    - liquidityAddress
    - marketingWallet
- **Supontis**
  - Initialize function is not protected from calling from other addresses. Everyone is able to call this function.

**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/IPancakeRouter01.sol	—————	1	96	5	3	1	48	
	contracts/IPancakePair.sol	—————	1	53	8	5	1	55	—————
	contracts/Dogeliens.sol	1	1	1039	888	786	15	429	
	contracts/IPancakeRouter02.sol	—————	1	44	6	4	1	16	
	contracts/PancakeCaller.sol	1	2	255	39	35	1	101	
	contracts/Supontis.sol	1	2	628	549	466	22	329	
	contracts/IPancakeFactory.sol	—————	1	20	7	4	1	19	—————
	contracts/Rocketize.sol	1	—————	908	773	671	14	380	
	<b>Totals</b>	<b>4</b>	<b>9</b>	<b>3043</b>	<b>2275</b>	<b>1974</b>	<b>56</b>	<b>1377</b>	

### Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalised lines of the source unit (e.g. normalises functions spanning multiple lines)
nSLOC	normalised 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

## Critical issues

**No critical issues**

## High issues

**No high issues**

## Medium issues

**No medium issues**

## Low issues

Issue	File	Type	Line	Description
#1	Supontis	Missing Zero Address Validation (missing-zero-check)	124, 125, 126, 123, 366, 374, 223, 357	Check that the address is not zero
#2	Supontis	State variable visibility is not set	41	It is best practice to set the visibility of state variables explicitly
#3	Supontis	Local variables shadowing	119, 120	Rename the local variables that shadow another component
#4	Supontis	Missing Events Arithmetic	136 142 144 145 146 149 150 151 154 155 156 297	Emit an event for critical parameter changes

## Informational issues

Issue	File	Type	Line	Description
-------	------	------	------	-------------

#1	Supontis	Error message is missing	388	Provide an error message for any require statement in the contract
#2	Supontis	NatSpec documentation missing	-	If you started to comment your code, also comment all other functions, variables etc.

## 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.

### 23. September 2022:

- Treasury contract was not provided to solidproof. Please DYOR here
- Owner can deploy a new version of the contract which can change any limit and give owner new privileges
- Read whole report and modifiers section 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>	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>NOT 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>	PASSED
<a href="#">SW C-11 5</a>	Authorization through tx.origin	<a href="#">CWE-477: Use of Obsolete Function</a>	PASSED
<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>	PASSED
<a href="#">SW C-11 3</a>	DoS with Failed Call	<a href="#">CWE-703: Improper Check or Handling of Exceptional Conditions</a>	PASSED
<a href="#">SW C-11 2</a>	Delegatecall to Untrusted Callee	<a href="#">CWE-829: Inclusion of Functionality from Untrusted Control Sphere</a>	PASSED
<a href="#">SW C-11 1</a>	Use of Deprecated Solidity Functions	<a href="#">CWE-477: Use of Obsolete Function</a>	PASSED
<a href="#">SW C-11 0</a>	Assert Violation	<a href="#">CWE-670: Always-Incorrect Control Flow Implementation</a>	PASSED
<a href="#">SW C-1 09</a>	Uninitialized Storage Pointer	<a href="#">CWE-824: Access of Uninitialized Pointer</a>	PASSED
<a href="#">SW C-1 08</a>	State Variable Default Visibility	<a href="#">CWE-710: Improper Adherence to Coding Standards</a>	NOT PASSED
<a href="#">SW C-1 07</a>	Reentrancy	<a href="#">CWE-841: Improper Enforcement of Behavioral Workflow</a>	PASSED
<a href="#">SW C-1 06</a>	Unprotected SELFDESTRUCT Instruction	<a href="#">CWE-284: Improper Access Control</a>	PASSED



<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>

*Solid  
Proofed*

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

  
MADE IN GERMANY