SOLIDProof

Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

v1.0 10. August, 2021

Audit

Security Assessment

04. September 2021

For

BINAPET

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	10. August 2021	• Layout project • Automated- /Manual-Security Testing • Summary
2.0	04. September 2021	• New Binapet contract

Network
Binance Smart Chain (BEP20)

Website
https://binapet.com/

Telegram
https://t.me/binapet
https://t.me/binapetchannel

Twitter
https://twitter.com/binapet

Instagram
https://instagram.com/binapetbsc?utm_medium=copy_link

Description

Binapet is a platform that integrates NFT games and decentralized yield farm applications. Joining Binapet not only entertains you but also generates a lot of profit.Our mission is to build a comprehensive platform of digital monsters that will enable millions of individuals to participate in the NFT and blockchain-based gaming world in a simple, creative, and enjoyable way.

Binapet will be the first ecosystem to combine the greatest aspects of gaming and digital collectibles, transforming it into the digital creatures universe. With Binapet, Players can use their pets to fight, collect, grow, and earn money.

Project Engagement

During the 8th of August 2021, **BinaPet 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. **BinaPet Team** provided Solidproof.io with access to their code repository and whitepaper.

Logo

Contract Link

v1.0

Binapet: https://bscscan.com/address/0xb26D2d67C7652f361b73552310dA5753041b3867#code

BinaPetBonus: https://bscscan.com/address/0x1d2b37255b02b7e512969ecbc1770b3b533807ae#code

BinaPetNFT: https://bscscan.com/address/0xba18456513bff2538f15a0c181141102963f10b3#code

BinaPetFarm: https://bscscan.com/address/0xe85f405d33689740a0042a0ec37079d9d92ddae3#code

BinaPetMarket: https://bscscan.com/address/0x9779eda1f8c683f31d72f5a709bdc9533f92fa59#code

v2.0

Binapet: https://bscscan.com/address/0x24D787e9B88Cb62D74e961C1C1d78E4ee47618E5#code

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)
Critical	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.
High	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 aspossible.
Medium	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.
Low	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.
Informational	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)

BinaPet

Imported packages:

v1.0

- OpenZeppelin
  - Address
  - Ownable
  - SafeMath
- Pancakeswap
  - PancakeFactory
  - PancakePair
  - PancakeRouter

v2.0

- TokenGuard added

BinaPetBonus

Imported packages:

- OpenZeppelin
  - Ownable
  - SafeMath

BinaPetFarm

Imported packages:

- OpenZeppelin
  - Ownable
  - SafeMath

BinaPetMarket

Imported packages:

- OpenZeppelin
  - Ownable
  - SafeMath

BinaPetNFT

Imported packages:

- OpenZeppelin
  - Ownable
  - SafeMath
  - Address

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/BinaPetFarm.sol	a025ca4c28be9ccc146356e559b2c95381b342f2
contracts/BinaPetMarket.sol	f88b566c92cf6586b94cde95715ab27465bba30e
contracts/BinaPet.sol	01fb44660d972bf67a327fb2900b27c1e3ddedc6
contracts/BinaPetNFT.sol	85e6dac7479ed900d31e93c0dd49f9f1674d0388
contracts/BinaPetBonus.sol	d378394cf9ccc486e3d2b9bcb65d812218e2261e

v2.0

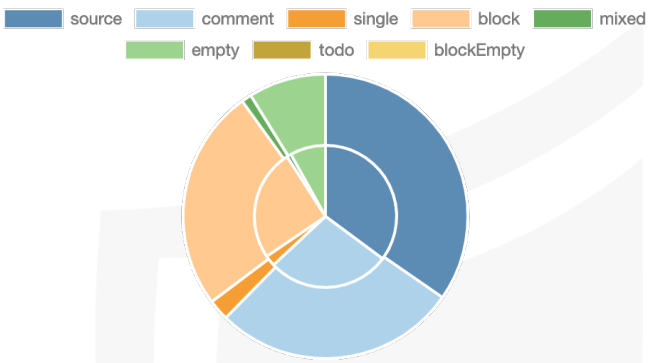
The audit v2.0 includes only the new contract Binapet

File Name	SHA-1 Hash
contracts/BinaPet.sol	9806fb550cf80bca42116929afe9f1405e0ff9e4

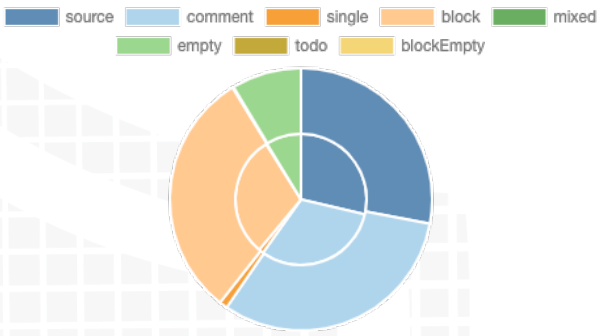
Metrics

Source Lines

v1.0

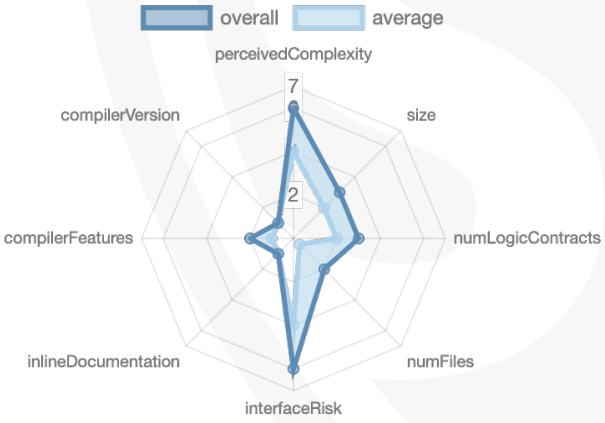


v2.0 (Binapet)

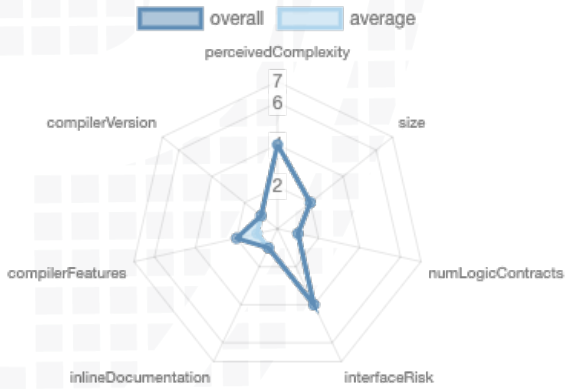


Risk Level

v1.0



v2.0 (Binapet)



Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	5	7	16	15
2.0	1	1	6	2

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	194	0
2.0	39	0

Version	External	Internal	Private	Pure	View
1.0	69	253	9	73	91
2.0	17	44	2	15	20

State Variables

Version	Total	Public
1.0	65	42
2.0	39	0

Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	`0.8.0`			`yes` (3 asm blocks)	
2.0	`0.8.0`			**** (0 asm blocks)	

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/Create/Create2
1.0	`yes`		`yes`	`yes`		
2.0	`yes`					

Scope of Work

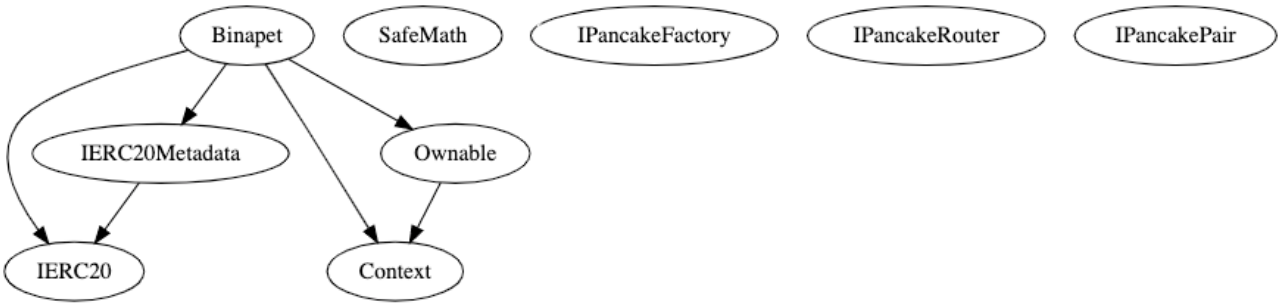
The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

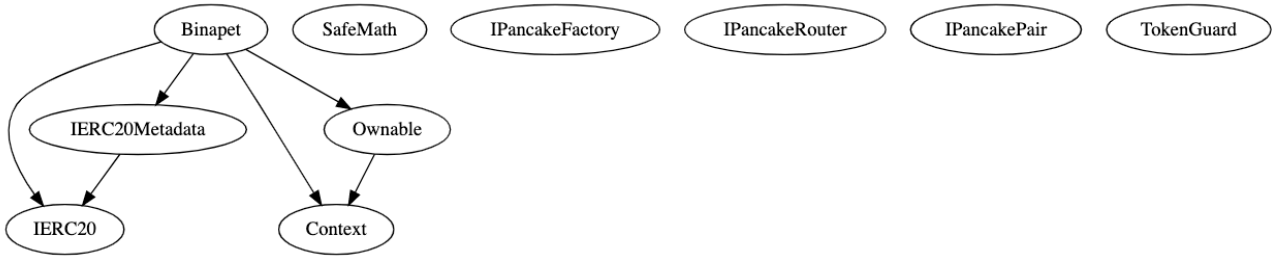
1. Correct implementation of Token standard
2. Deployer cannot mint any new tokens
3. Deployer cannot burn or lock user funds
4. Deployer cannot pause the contract
5. Overall checkup (Smart Contract Security)

Inheritance Graph

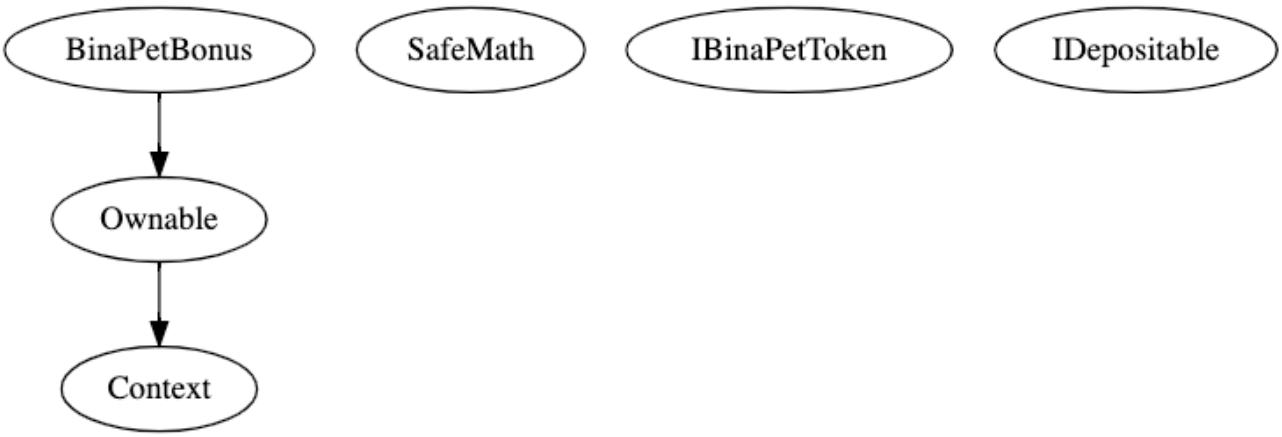
BinaPet v1.0



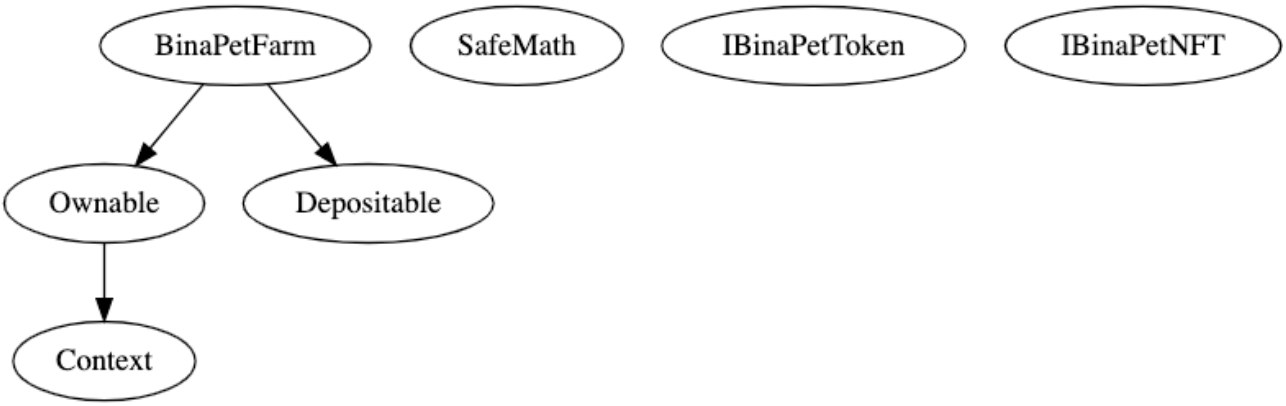
v2.0



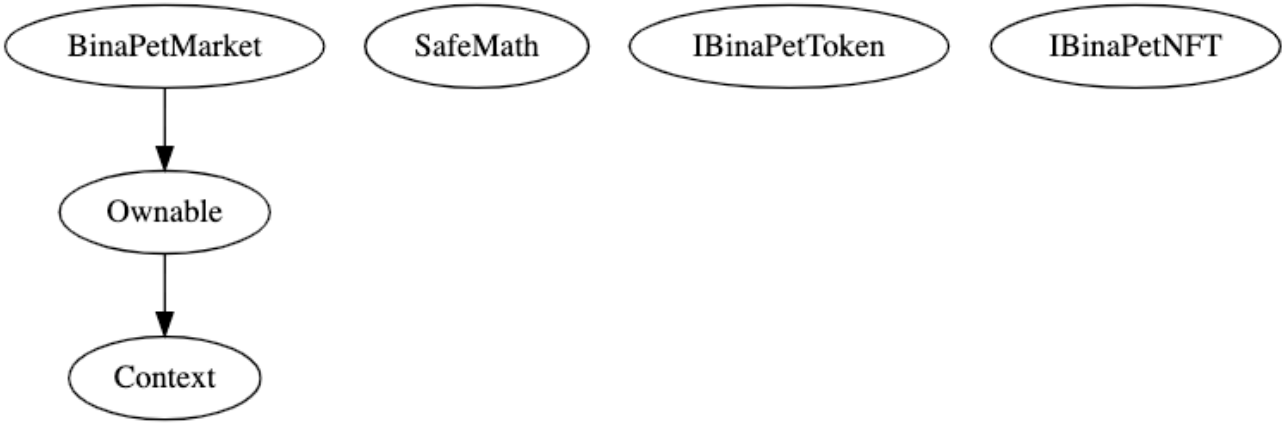
BinaPetBonus



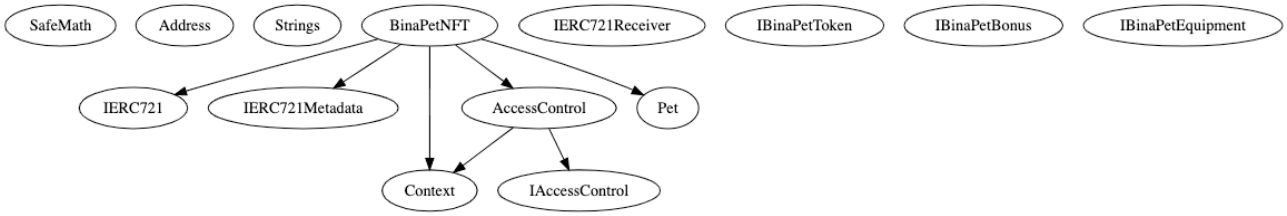
BinaPetFarm



BinaPetMarket



BinaPetNFT


SafeMath
Address
Strings
BinaPetNFT
IERC721Receiver
IBinaPetToken
IBinaPetBonus
IBinaPetEquipment
IERC721
IERC721Metadata
AccessControl
Pet
Context
IAccessControl

Verify Claims

Correct implementation of Token standard

Tested	Verified
✓	✓

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	✓	✓	✓

Optional implementations

Function	Description	Exist	Tested	Verified
renounceOwnership	Owner renounce ownership for more trust	✓	✓	⚑

Deployer cannot mint any new tokens

Tested	Deployer cannot mint	File	Comment
✓	✓	Main	Line: -

Max / Total Supply:

```
constructor() {
    _mint(msg.sender, _initSupply);

    taxAddress = payable(msg.sender);

    IPancakeRouter _router = IPancakeRouter(0xECC5428A66808FC40A464e5B3F4D265Df985E3E8); //for test
    //IPancakeRouter _router = IPancakeRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);


    pairAddress = IPancakeFactory(_router.factory())
        .createPair(address(this), _router.WETH());

    // set the rest of the contract variables
    routerAddress = address(_router);

    _isExcludedFromFee[owner()] = true;
}
```

```
function _mint(address account, uint256 amount) internal virtual {
        require(account != address(0), "ERC20: mint to the zero address");

        _beforeTokenTransfer(address(0), account, amount);

        _totalSupply = _totalSupply.add(amount);
        _balances[account] = _balances[account].add(amount);
        emit Transfer(address(0), account, amount);
    }
```

Deployer cannot burn or lock user funds

Name	Tested	Exist	Verified
Deployer cannot lock	✓	✓	✓
Deployer cannot burn	✓	✓	✓

Deployer cannot pause the contract

Tested	Verified	Deployer cannot pause
✓	✓	✓

1. approve →
2. decreaseAllowance →
3. excludeFromFee →
4. includeInFee →
5. increaseAllowance →
6. renounceOwnership →
7. setTax →
8. setTaxAddress →
9. transfer →
10. transferFrom →
11. transferOwnership →

Browse source code

Overall checkup (Smart Contract Security)

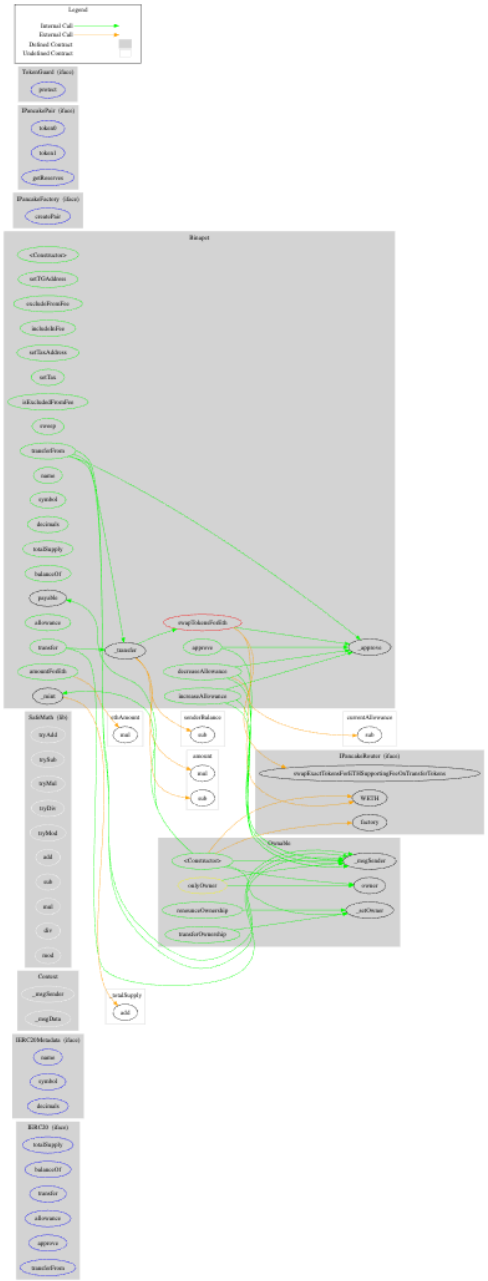
Tested	Verified
✓	✓

Legend

Attribute	Symbol
Verfified / Checked	✓
Partly Verified	⚑
Unverified / Not checked	✗

CallGraph

v2.0 (Binapet)



Source Units in Scope

v1.0

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/BinaPetFarm.sol	6	1	648	575	312	200	280	
	contracts/BinaPetMarket.sol	5	1	487	409	186	187	157	
	contracts/BinaPet.sol	4	5	765	647	257	379	192	
	contracts/BinaPetNFT.sol	8	7	1992	1679	934	757	693	
	contracts/BinaPetBonus.sol	4	2	391	357	149	174	99	
	Totals	**27**	**16**	**4283**	**3667**	**1838**	**1697**	**1421**	

v2.0

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	contracts/BinaPet.sol	4	6	825	697	292	384	237	
	Totals	**4**	**6**	**825**	**697**	**292**	**384**	**237**	

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 found -

High issues

- no high issues found -

Medium issues

- no medium issues found -

Low issues

v1.0

- no low issues found -

V2.0

Issue	File	Type	Line	Description
#1	Binapet	Missing Zero Address Validation (missing-zero-check)	524, 512	Check that the address is not zero

Informational issues

- no informational issues found -

Audit Comments

10. August 2021:

- There is still an owner (Owner still has not renounced ownership)

04. September 2021:

- New Binapet contract were referred
- Following changes were made
  - IPancakeRouter interface
    - Added function swapExactTokensForETCSupportingFeeOnTransferTokens
  - Added
    - ITokenGuard interface
    - new SwapTokensForETH event
    - New address tgAddress
    - setTGAddress function
    - New sweep function
    - Implemented swapTokensForEth function from interface
- We have no insight in the protect function in TokenGuard because it comes from outside

SWC Attacks

ID	Title	Relationships	Status
SWC-136	Unencrypted Private Data On-Chain	CWE-767: Access to Critical Private Variable via Public Method	PASSED
SWC-135	Code With No Effects	CWE-1164: Irrelevant Code	PASSED
SWC-134	Message call with hardcoded gas amount	CWE-655: Improper Initialization	PASSED
SWC-133	Hash Collisions With Multiple Variable Length Arguments	CWE-294: Authentication Bypass by Capture-replay	PASSED
SWC-132	Unexpected Ether balance	CWE-667: Improper Locking	PASSED
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	PASSED
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	PASSED
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	PASSED
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	PASSED

SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	PASSED
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	PASSED
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	PASSED
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	PASSED
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	PASSED
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	PASSED
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	PASSED
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	PASSED
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	PASSED
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	PASSED

SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	PASSED
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	PASSED
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	PASSED
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	PASSED
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	PASSED
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	PASSED
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	PASSED
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	PASSED

SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	PASSED
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	PASSED
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	PASSED
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	PASSED
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	PASSED
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY