Shiba inu(SHIB)
Smartcontract Security
Audit Report

2022. 02

From SCOPE

https://blosafe.com

SCOPE

2022. 02

Document History

Date	Name	History
2022.02	Blosafe	Initial

1. Project outline

1.1. Purpose

The purpose of this inspection is to conduct a security audit on the [Shiba inu Name] Smartcontract to discover potential hacking weaknesses, analyze the cause, and respond

1.2. Target

The subjects of this inspection are as follows.

No	Category	Addr	Memo
1	Smartcontract	0x95aD61b0a150d79219dCF64E1E6Cc01f0B64C4cE	ETH Mainnet

1.3. Schedule

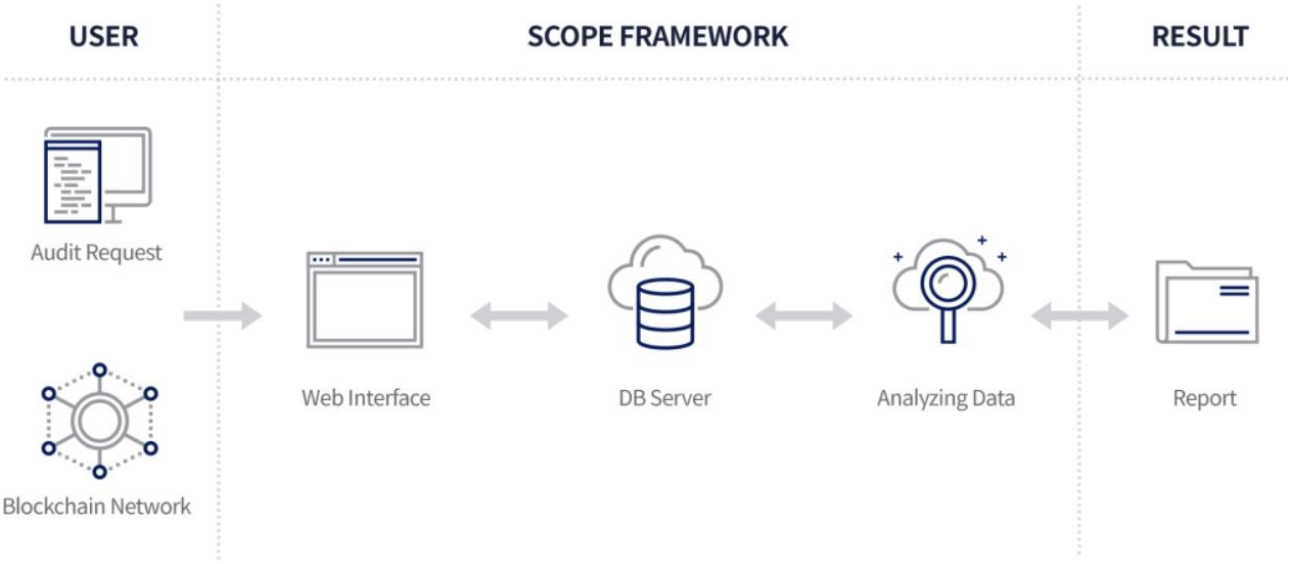
Work	Detail	Timeline	Memo
business consultation	Build Environment	1 day	
Audit	Smartcontract static auditing	2 days	
	Smartcontract Dynamic Auditing	3 days	
Report / review	Report	1 day	
	Review	1 day	

1.4. Environment

업무 구분	Name	Platform	Memo
Audit	Scope Audit	SaaS	

2. Process

2.1. Process Detail



2.2. Check List

No	Detector	What it Detects	Impact	Confidence
1	`abiencoderv2-array`	Storage abiencoderv2 array	High	High
2	`array-by-reference`	Modifying storage array by value	High	High
3	`incorrect-shift`	The order of parameters in a shift instruction is incorrect.	High	High
4	`multiple-constructors`	Multiple constructor schemes	High	High
5	`name-reused`	Contract's name reused	High	High
6	`public-mappings-nested`	Public mappings with nested variables	High	High
7	`rtlo`	Right-To-Left-Override control character is used	High	High
8	`shadowing-state`	State variables shadowing	High	High
9	`suicidal`	Functions allowing anyone to destruct the contract	High	High

10	uninitialized-state	Uninitialized state variables	High	High
11	uninitialized-storage	Uninitialized storage variables	High	High
12	unprotected-upgrade	Unprotected upgradeable contract	High	High
13	arbitrary-send	Functions that send Ether to arbitrary destinations	High	Medium
14	controlled-array-length	Tainted array length assignment	High	Medium
15	controlled-delegatecall	Controlled delegatecall destination	High	Medium
16	delegatecall-loop	Payable functions using delegatecall inside a loop	High	Medium
17	msg-value-loop	msg.value inside a loop	High	Medium
18	reentrancy-eth	Reentrancy vulnerabilities (theft of ethers)	High	Medium
19	storage-array	Signed storage integer array compiler bug	High	Medium
20	unchecked-transfer	Unchecked tokens transfer	High	Medium
21	weak-prng	Weak PRNG	High	Medium
22	enum-conversion	Detect dangerous enum conversion	Medium	High
23	erc20-interface	Incorrect ERC20 interfaces	Medium	High
24	erc721-interface	Incorrect ERC721 interfaces	Medium	High
25	incorrect-equality	Dangerous strict equalities	Medium	High
26	locked-ether	Contracts that lock ether	Medium	High
27	mapping-deletion	Deletion on mapping containing a structure	Medium	High
28	shadowing-abstract	State variables shadowing from abstract contracts	Medium	High
29	tautology	Tautology or contradiction	Medium	High
30	write-after-write	Unused write	Medium	High
31	boolean-cst	Misuse of Boolean constant	Medium	Medium
32	constant-function-asm	Constant functions using assembly code	Medium	Medium

SCOPE

33	constant-function-state	Constant functions changing the state	Medium	Medium
34	divide-before-multiply	Imprecise arithmetic operations order	Medium	Medium
35	reentrancy-no-eth	Reentrancy vulnerabilities (no theft of ethers)	Medium	Medium
36	reused-constructor	Reused base constructor	Medium	Medium
37	tx-origin	Dangerous usage of tx.origin	Medium	Medium
38	unchecked-lowlevel	Unchecked low-level calls	Medium	Medium
39	unchecked-send	Unchecked send	Medium	Medium
40	uninitialized-local	Uninitialized local variables	Medium	Medium
41	unused-return	Unused return values	Medium	Medium
42	incorrect-modifier	Modifiers that can return the default value	Low	High
43	shadowing-builtin	Built-in symbol shadowing	Low	High
44	shadowing-local	Local variables shadowing	Low	High
45	uninitialized-fptr-cst	Uninitialized function pointer calls in constructors	Low	High
46	variable-scope	Local variables used prior their declaration	Low	High
47	void-cst	Constructor called not implemented	Low	High
48	calls-loop	Multiple calls in a loop	Low	Medium
49	events-access	Missing Events Access Control	Low	Medium
50	events-maths	Missing Events Arithmetic	Low	Medium
51	incorrect-unary	Dangerous unary expressions	Low	Medium
52	missing-zero-check	Missing Zero Address Validation	Low	Medium
53	reentrancy-benign	Benign reentrancy vulnerabilities	Low	Medium
54	reentrancy-events	Reentrancy vulnerabilities leading to out-of-order Events	Low	Medium
55	timestamp	Dangerous usage	Low	Medium

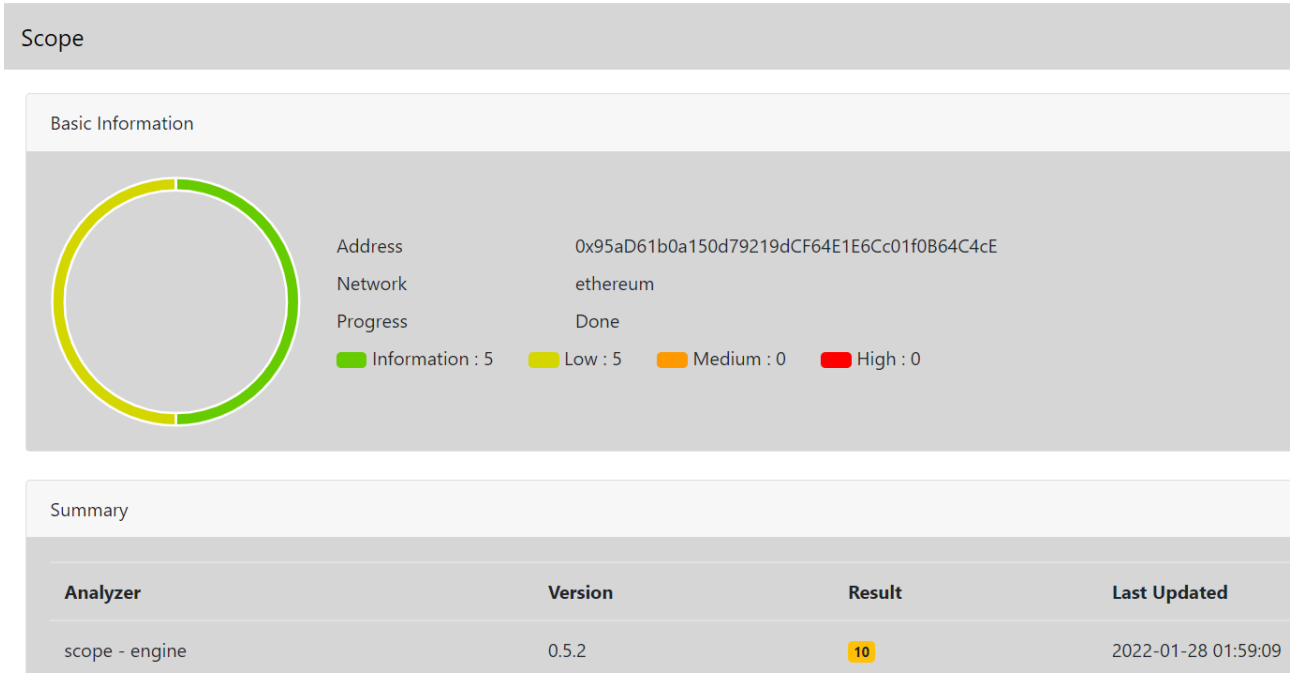
SCOPE

		of block.timestamp		
56	assembly	Assembly usage	Informational	High
57	assert-state-change	Assert state change	Informational	High
58	boolean-equal	Comparison to boolean constant	Informational	High
59	deprecated-standards	Deprecated Solidity Standards	Informational	High
60	erc20-indexed	Un-indexed ERC20 event parameters	Informational	High
61	function-init-state	Function initializing state variables	Informational	High
62	low-level-calls	Low level calls	Informational	High
63	missing-inheritance	Missing inheritance	Informational	High
64	naming-convention	Conformity to Solidity naming conventions	Informational	High
65	pragma	If different pragma directives are used	Informational	High
66	redundant-statements	Redundant statements	Informational	High
67	solc-version	Incorrect Solidity version	Informational	High
68	unimplemented-functions	Unimplemented functions	Informational	High
69	unused-state	Unused state variables	Informational	High
70	costly-loop	Costly operations in a loop	Informational	Medium
71	dead-code	Functions that are not used	Informational	Medium
72	reentrancy-unlimited-gas	Reentrancy vulnerabilities through send and transfer	Informational	Medium
73	similar-names	Variable names are too similar	Informational	Medium
74	too-many-digits	Conformance to numeric notation best practices	Informational	Medium
75	constable-states	State variables that could be declared constant	Optimization	High
76	external-function	Public function that could be declared external	Optimization	High

SCOPE

3. Summary of results

3.1. Result



[Passed]

[Shiba inu] As a result of the Smartcontract security audit, a total of 10 vulnerabilities were found, among which 0 vulnerabilities of 'high', 0 of 'medium' vulnerabilities, 5 of 'low' vulnerabilities, and 'information' ratings were found.

4. Detailed results

4.1. Smartcontract

```
/**
 *Submitted for verification at Etherscan.io on 2021-02-26
*/

/**
 *Submitted for verification at Etherscan.io on 2019-08-02
*/

// File: contracts₩open-zeppelin-contracts₩token₩ERC20₩IERC20.sol

pragma solidity ^0.5.0;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP. Does not include
 * the optional functions; to access them see `ERC20Detailed`.
 */
interface IERC20 {
    /**
     * @dev Returns the amount of tokens in existence.
     */
    function totalSupply() external view returns (uint256);
```

```
/**
 * @dev Returns the amount of tokens owned by `account`.
 */
function balanceOf(address account) external view returns (uint256);

/**
 * @dev Moves `amount` tokens from the caller's account to `recipient`.
 *
 * Returns a boolean value indicating whether the operation succeeded.
 *
 * Emits a `Transfer` event.
 */
function transfer(address recipient, uint256 amount) external returns (bool);

/**
 * @dev Returns the remaining number of tokens that `spender` will be
 * allowed to spend on behalf of `owner` through `transferFrom`. This is
 * zero by default.
 *
 * This value changes when `approve` or `transferFrom` are called.
 */
function allowance(address owner, address spender) external view returns (uint256);
```

.

SKIP

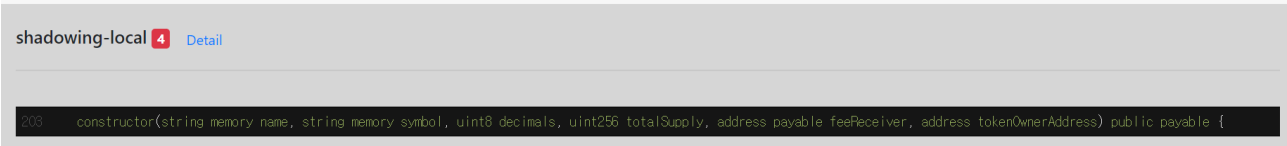
SKIP

```
_mint(tokenOwnerAddress, totalSupply);


    // pay the service fee for contract deployment
    feeReceiver.transfer(msg.value);
}


/**
 * @dev Burns a specific amount of tokens.
 * @param value The amount of lowest token units to be burned.
 */
function burn(uint256 value) public {
    _burn(msg.sender, value);
}


// optional functions from ERC20 stardard


/**
 * @return the name of the token.
 */
function name() public view returns (string memory) {
    return _name;
}


/**
 * @return the symbol of the token.
```

```
 */
function symbol() public view returns (string memory) {
    return _symbol;
}

/**
 * @return the number of decimals of the token.
 */
function decimals() public view returns (uint8) {
    return _decimals;
}
```

4.2. Vulnerability

4.2.1. State variable shadowing



Configuration

Check: shadowing-state
Severity: High
Confidence: High

Description

Detection of state variables shadowed.

Exploit Scenario:

```
contract BaseContract{
    address owner;
```

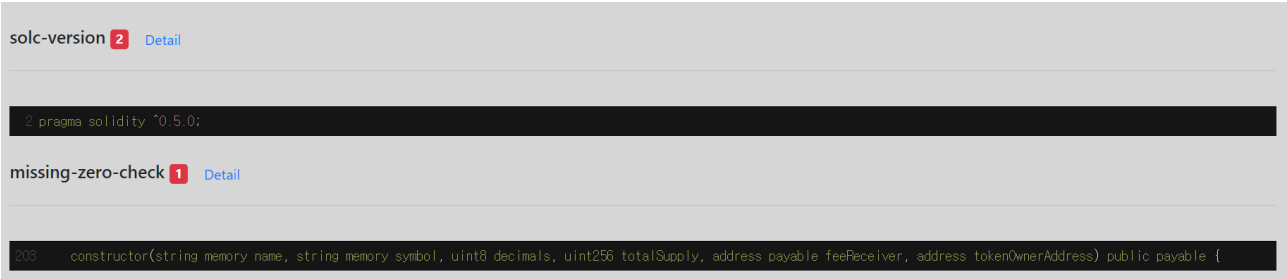
```
    modifier isOwner(){
        require(owner == msg.sender);
        _;
    }

}

contract DerivedContract is BaseContract{
    address owner;

    constructor(){
        owner = msg.sender;
    }

    function withdraw() isOwner() external{
        msg.sender.transfer(this.balance);
    }
}
```

owner of BaseContract is never assigned and the modifier isOwner does not work.

Recommendation

Remove the state variable shadowing.

4.2.2. Missing zero address validation



Configuration

Check: missing-zero-check
Severity: Low
Confidence: Medium

Description

Detect missing zero address validation.

Recommendation

Check that the address is not zero.

4.2.3. Dead code

dead-code 4 Detail

```
185     function _burnFrom(address account, uint256 amount) internal {
186         _burn(account, amount);
187         _approve(account, msg.sender, _allowances[account][msg.sender].sub(amount));
188     }
```

```
72     function div(uint256 a, uint256 b) internal pure returns (uint256) {
73                 require(b > 0, "SafeMath: division by zero");
74         uint256 c = a / b;
75
76         return c;
77     }
```

```
80     function mod(uint256 a, uint256 b) internal pure returns (uint256) {
81         require(b != 0, "SafeMath: modulo by zero");
82         return a % b;
83     }
```

Configuration

Check: dead-code
Severity: Informational
Confidence: Medium

Description

Functions that are not sued.

Recommendation

Remove unused functions.