CERTIK

Security Assessment

Solv Protocol

Sept 30th, 2021

Table of Contents

Summary

This report has been prepared for SOLV FOUNDATION LTD. to discover issues and vulnerabilities in the source code of the Solv Protocol project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Additionally, this audit is based on a premise that all external smart contracts are implemented safely.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Solv Protocol
Platform	Ethereum, BSC
Language	Solidity
Codebase	1. https://github.com/solv-finance/solv-v2-helper 2. https://github.com/solv-finance/solv-v2-market 3. https://github.com/solv-finance/solv-v2-voucher
Commit	1. a6568b839e963a265bd80fc21439a52e5035b5dc 2. f54e1934a2b4e25ee8ba1be5b3c08a1ccdb4050f 3. 8821fcc8bd47d1bae6fb0071ebc36312d39ea041

Audit Summary

Delivery Date	Sept 30, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
● Critical	0	0	0	0	0	0
● Major	1	0	0	1	0	0
● Medium	0	0	0	0	0	0
● Minor	2	0	0	2	0	0
● Informational	15	0	0	15	0	0
● Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
ERC	solv-v2-helper/helpers/ERC20TransferHelper.sol	b958dbe1a3a2964e73583f6fb052a6022d6e199a7bcb1252d63877219dd06f37
EAL	solv-v2-helper/helpers/EthAddressLib.sol	dd826f55dc8cb3dede33df21a224ba1944b4ba7b786739c3788d638bab889e31
VNF	solv-v2-helper/helpers/VNFTTransferHelper.sol	92ce7064f8e39d85f3295bcc7c4653d1a7b52159e51dafc9317b03afd109970e
AUP	solv-v2-helper/proxy/contracts/AdminUpgradeabilityProxy.sol	7d55b427f3907870f9bd393a753e512d1502f43ebfd727f5b8a420623fd51510
PCK	solv-v2-helper/proxy/contracts/Proxy.sol	a29edf43aa6bfdd06a0de5f2f128aab9ee931a90a9da36e3fb622a464e1e13d6
PAC	solv-v2-helper/proxy/contracts/ProxyAdmin.sol	a6f908c5450cafa6602fbe28c48afad8aa28b8152e0f35129aa4dd4dd82ed115
UPC	solv-v2-helper/proxy/contracts/UpgradeabilityProxy.sol	a648facee4d7f4ba362dc8890ef771e713515eb14dfa1f0ccabd25a18b3a22f6
ISC	solv-v2-market/packages/solv-market/contracts/interface/external/ISolver.sol	d56603d00f9e6d04125290ebe2fb63e5fc2730d378d13596ddee08b073f74eee
IUC	solv-v2-market/packages/solv-market/contracts/interface/external/IUnderlyingContainer.sol	854a377aab42a69254ab8494fa75f63b6662f0f5daca3bb77515361674254 28a
IVN	solv-v2-market/packages/solv-market/contracts/interface/external/IVNFT.sol	c16b7b0552e69aab2c04f488222e87d10a7ff9611af8910db495ddaa337af885
ISI	solv-v2-market/packages/solv-market/contracts/interface/ISolvICMarket.sol	1c3b4492ad65a96b236cd93f8d6b21eac80190462bfc2cd6f4e86e752de85099
PMC	solv-v2-market/packages/solv-market/contracts/PriceManager.sol	15da4a24e0ee6acbab85f01b5cd7439c37e5ae50e43c00b3e09b1a58635c29c4
SMU	solv-v2-market/packages/solv-market/contracts/SafeMathUpgradeable128.sol	410acc682cec331d746e46990660f1ee88877b371566f1f5e228ee124586b88d
SIC	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol	54e8e2faa74abd0d76444b8a5daa9eae71817bf36492e4054efe5b2e8dde42f2

ID	File	SHA256 Checksum
SOC	solv-v2-voucher/packages/solv-token/contracts/SOLV.sol	69343e6108aed6c88578c39ee93de08789ceb3f0a4958cd9b4edec4e0e82a5ae
IVF	solv-v2-voucher/packages/solv-vnft-core/contracts/interface/IVNFT.sol	35f976cb8f06580129f049fd13d4b2eed370e375fec328c2a04617943f07d848
ALC	solv-v2-voucher/packages/solv-vnft-core/contracts/library/AssetLibrary.sol	c405c56af0b716004dd2bb1c3b095d4b2a4aa461f60552dbff70f1fa01b81145
VNT	solv-v2-voucher/packages/solv-vnft-core/contracts/VNFTCore.sol	763adfdb809e93f06da478479ebc14e125c3507e3dcd42c2edda3c243ee90379
IIC	solv-v2-voucher/packages/solv-voucher/contracts/interface/IICToken.sol	2e74ffdb4dae73263bbbad781157d9dc9257d5fd0439cf970cb8e6433f058a34
IUK	solv-v2-voucher/packages/solv-voucher/contracts/interface/IUnderlyingContainer.sol	0eb9a398eaecef87271a07ee813361df0de69d09c21f071b288ab9fabad1b326
IVT	solv-v2-voucher/packages/solv-voucher/contracts/interface/IVNFTErc20Container.sol	88ed486a0f50c9a04b6a3534ae6bd3f10e86b6c0e42810670b9f22f9d65c4189
IVP	solv-v2-voucher/packages/solv-voucher/contracts/interface/IVestingPool.sol	23ca5d8cc161cc7d2fcdeb4295445a1ee246ca4c77b824d4684fa85945d593d9
VLC	solv-v2-voucher/packages/solv-voucher/contracts/library/VestingLibrary.sol	befb9821a13d7fe6d35a95b2d814f4d8128758597be53dc98bdca3d5bc4ffcd9
ICT	solv-v2-voucher/packages/solv-voucher/contracts/ICToken.sol	2c026c52e6c5e3856ba208179d234dbb16dd5d29bf35a389c2af4665164fd391
VPC	solv-v2-voucher/packages/solv-voucher/contracts/VestingPool.sol	d0504a932b4eed7cfcf24d64d5ec6b63e70846836e3faf6fbd70ff422021f96b
ISK	solv-v2-voucher/packages/solver/contracts/interface/ISolver.sol	9ad79ab5e12bd8889f9f0a9c0aa0926b479b52ef88ee6df9d77601f4d59f2daa
SCK	solv-v2-voucher/packages/solver/contracts/Solver.sol	09a8eda6b27c25acc6158d30897f19f59c5e25e5618a1bf5b9f11729b76ac115

Understandings

Overview

Solv mainly provides functions such as splitting and merging of Finance NFT. In this period, it realizes mint, transfer, merge, split and corresponding market of investment shares. The market mainly includes the functions of pending orders, canceling orders, and buying orders. The investment share is to lock the Token of the project party into the contract, and specify the lock time, which can be unlocked in a linear, phased or one-time basis. The market is OTC, and sellers can choose fixed-price and Dutch auctions for placing orders, and buyers can purchase part of it.

Privileged Functions

The contract contains the following privileged functions that are restricted by some modifiers. They are used to modify the contract configurations and address attributes. We grouped these functions below:

The `onlyOwner` modifier:

Contract `ProxyAdmin`:

- changeProxyAdmin(AdminUpgradeabilityProxy proxy, address newAdmin)
- upgrade(AdminUpgradeabilityProxy proxy, address implementation)
- upgradeAndCall(AdminUpgradeabilityProxy proxy, address implementation, bytes memory data)

The `onlyAdmin` modifier:

Contract `ICToken`:

- setContractURI(string memory uri_)
- setBaseURI(string memory uri_)
- upgradeAndCall(AdminUpgradeabilityProxy proxy, address implementation, bytes memory data)
- _setSolver(ISolver newSolver_)
- _setVestingPool(IVestingPool newVestingPool_)

Contract `VestingPool`:

- _setManager(address newManager_)
- _setBaseImageURI(string memory uri_)
- _setBaseExternalURI(string memory uri_)
- _setSolver(ISolver newSolver_)

- _setVestingPool(IVestingPool newVestingPool_)

Contract `Solver`:

- _setTransferGuuardianPause(address product, bool enable)
- _setDepositGuuardianPause(address product, bool enable)
- _setWithdrawGuuardianPause(address product, bool enable)
- _setConvertUnsafeTransferContracts(address product, bool enable)
- _setRejectUnsafeTransferContracts(address product, bool enable)

Contract `SolvICMarket`:

- _setManager(address newManager_)
- _addMarket(address icToken_, uint64 precision_, uint8 feePayType_, uint8 feeType_, uint128 feeAmount_, uint16 feeRate_)
- _removeMarket(address icToken_)
- _setCurrency(address currency_, bool enable_)
- _setRepoFeeRate(uint16 newRepoFeeRate_)
- _withdrawFee(address icToken_, uint256 reduceAmount_)
- setAllowAddressManager(address icToken_, address[] calldata managers_, bool resetExisting_)
- _setSolver(ISolver newSolver_)

The `onlyAllowAddressManager` modifier:

Contract `SolvICMarket`:

- _addAllowAddress(address icToken_, address[] calldata addresses_, bool resetExisting_)
- _removeAllowAddress(address icToken_, address[] calldata addresses_)

The `onlyManager` modifier:

Contract `VestingPool`:

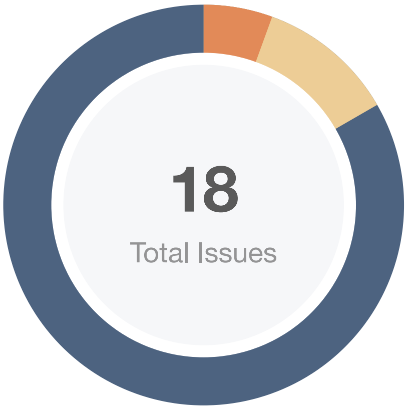
- mint(uint8 claimType_, address minter_, uint256 tokenId_, uint64 term_, uint256 amount_, uint64[] calldata maturities_, uint32[] calldata percentages_, string memory originalInvestor_)
- recharge(address recharger_, address owner_, uint256 tokenId_, uint256 amount_)
- claim(address payable payee, uint256 tokenId, uint256 amount)
- transferVesting(address from_, uint256 tokenId_, address to_, uint256 targetTokenId_, uint256 transferUnits_)
- splitVesting(address owner_, uint256 tokenId_, uint256 newTokenId_, uint256 splitUnits_)
- mergeVesting(address owner_, uint256 tokenId_, uint256 targetTokenId_)

The `ifAdmin` role:

Contract `AdminUpgradeabilityProxy`:

- admin()
- implementation()
- changeAdmin(address newAdmin)
- upgradeTo(address newImplementation)
- upgradeToAndCall(address newImplementation, bytes calldata data)

Findings



🟥	**Critical**	**0** (0.00%)
🟧	**Major**	**1** (5.56%)
🟨	**Medium**	**0** (0.00%)
🟨	**Minor**	**2** (11.11%)
🟦	**Informational**	**15** (83.33%)
🟩	**Discussion**	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Centralization Risk	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged
AUP-01	Missing Input Validation	Logical Issue	● Informational	ⓘ Acknowledged
ICT-01	Typos in the contract	Coding Style	● Informational	ⓘ Acknowledged
ICT-02	Function Visibility Optimization	Gas Optimization	● Informational	ⓘ Acknowledged
PAC-01	Function Visibility Optimization	Gas Optimization	● Informational	ⓘ Acknowledged
SCK-01	Typos in the contract	Coding Style	● Informational	ⓘ Acknowledged
SCK-02	Function Visibility Optimization	Gas Optimization	● Informational	ⓘ Acknowledged
SCK-03	Missing Emit Events	Coding Style	● Informational	ⓘ Acknowledged
SIC-01	Lack of input validation	Logical Issue	● Informational	ⓘ Acknowledged
SIC-02	Boolean Equality Optimization	Coding Style	● Informational	ⓘ Acknowledged
SIC-03	Function Visibility Optimization	Gas Optimization	● Informational	ⓘ Acknowledged
SIC-04	Missing Input Validation	Logical Issue	● Informational	ⓘ Acknowledged
SIC-05	Strengthen Transfer Security	Logical Issue	● Minor	ⓘ Acknowledged
SIC-06	Missing Emit Events	Logical Issue	● Informational	ⓘ Acknowledged
VNT-01	Missing add `targetTokenId_` to `_slotTokens`	Logical Issue	● Minor	ⓘ Acknowledged

ID	Title	Category	Severity	Status
VNT-02	Optimization For Function `_burnUnits()`	Logical Issue	● Informational	ⓘ Acknowledged
VPC-01	Boolean Equality Optimization	Coding Style	● Informational	ⓘ Acknowledged
VPC-02	Function Visibility Optimization	Gas Optimization	● Informational	ⓘ Acknowledged

GLOBAL-01 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	● Major	Global	ⓘ Acknowledged

Description

In the contract `AdminUpgradeabilityProxy`, the role `admin` has the authority over the following function:

- admin()
- implementation()
- changeAdmin()
- upgradeTo()
- upgradeToAndCall()

In the contract `ProxyAdmin`, the role `owner` has the authority over the following function:

- changeProxyAdmin()
- upgrade()
- upgradeAndCall()

In the contract `ICToken`, the role `admin` has the authority over the following function:

- setContractURI()
- setBaseURI()
- _setSolver()
- _setVestingPool()

In the contract `VestingPool`, the role `admin` or role `manager` has the authority over the following function:

- _setManager()
- _setBaseImageURI()
- _setBaseExternalURI()
- mint()
- recharge()
- claim()
- transferVesting()
- splitVesting()
- mergeVesting()

In the contract `SolvICMarket`, the role `admin` or role `allowAddressManagers` has the authority over the following function:

- _addMarket()
- _removeMarket()
- _setCurrency()
- _setRepoFeeRate()
- _withdrawFee()
- _addAllowAddress()
- _removeAllowAddress()
- setAllowAddressManager()
- _setSolver()

In the contract `SOLV`, it will mint `SOLV` tokens to `minter` when deploying this contract.

Any compromise to these accounts may allow the hacker to manipulate the project through these functions.

Recommendation

We advise the client to carefully manage the `admin/manager/owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at the different levels in terms of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

The development team responded that this issue will not be revised for the time being. Later, according to the situation, the management authority will be transferred to the timelock contract or voting mechanism, and finally handed over to the community.

AUP-01 | Missing Input Validation

Category	Severity	Location	Status
Logical Issue	● Informational	solv-v2-helper/proxy/contracts/AdminUpgradeabilityProxy.sol: 25	ⓘ Acknowledged

Description

The given input is missing the sanity check for non-zero address in the aforementioned line.

Recommendation

We recommend adding the check for the passed-in values to prevent unexpected error as below: constructor():

```
require(_initAdmin != address(0), "_initAdmin address cannot be 0");
```

Alleviation

No alleviation.

ICT-01 | Typos in the contract

Category	Severity	Location	Status
Coding Style	● Informational	solv-v2-voucher/packages/solv-voucher/contracts/ICToken.sol: 86 ~87	ⓘ Acknowledged

Description

There are several typos in these contracts. Contract:`Solver`

1. `_setTransferGuuardianPause` should be `_setTransferGuardianPause`.
2. `_setDepositGuuardianPause` should be `_setDepositGuardianPause`.
3. `_setWithdrawGuuardianPause` should be `_setWithdrawGuardianPause`.

Contract:`ICToken`

1. `hodler` should be `holder`.

Recommendation

We recommend correcting all typos in the contract.

Alleviation

The development team responded that they will fix this issue in the next version.

ICT-02 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	● Informational	solv-v2-voucher/packages/solv-voucher/contracts/ICToken.sol: 302, 320, 486, 494	ⓘ Acknowledged

Description

The following functions are declared as `public` and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

- `transferFrom()` in L302
- `transferFrom()` in L320
- `_setSolver()` in L486
- `_setVestingPool` in L494

Recommendation

We advise that the functions' visibility specifiers are set to `external` and the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

Alleviation

No alleviation.

PAC-01 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	● Informational	solv-v2-helper/proxy/contracts/ProxyAdmin.sol: 45, 54, 67	ⓘ Acknowledged

Description

The following functions are declared as `public` and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

- `changeProxyAdmin()` in L45
- `upgrade()` in L54
- `upgradeAndCall()` in L67

Recommendation

We advise that the functions' visibility specifiers are set to `external` and the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

Alleviation

No alleviation.

SCK-01 | Typos in the contract

Category	Severity	Location	Status
Coding Style	● Informational	solv-v2-voucher/packages/solver/contracts/Solver.sol: 32, 39, 46	ⓘ Acknowledged

Description

There are several typos in these contracts. Contract:`Solver`

1. `_setTransferGuuardianPause` should be `_setTransferGuardianPause`.
2. `_setDepositGuuardianPause` should be `_setDepositGuardianPause`.
3. `_setWithdrawGuuardianPause` should be `_setWithdrawGuardianPause`.

Contract:`ICToken`

1. `hodler` should be `holder`.

Recommendation

We recommend correcting all typos in the contract.

Alleviation

The development team responded that they will fix this issue in the next version.

SCK-02 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	● Informational	solv-v2-voucher/packages/solver/contracts/Solver.sol: 24, 32, 39, 46, 53, 60, 411, 424	ⓘ Acknowledged

Description

The following functions are declared as `public` and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

- `initialize()` in L24
- `_setTransferGuuardianPause()` in L32
- `_setDepositGuuardianPause()` in L39
- `_setWithdrawGuuardianPause()` in L46
- `_setConvertUnsafeTransferContracts()` in L53
- `_setRejectUnsafeTransferContracts()` in L60
- `_setPendingAdmin()` in L411
- `_acceptAdmin()` in L424

Recommendation

We advise that the functions' visibility specifiers are set to `external` and the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

Alleviation

No alleviation.

SCK-03 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	● Informational	solv-v2-voucher/packages/solver/contracts/Solver.sol: 32, 39, 46, 53, 60	ⓘ Acknowledged

Description

In contract `Solver`, there are numerous functions that can change state variables. However, these functions do not emit events to pass the changes out of chain.

Recommendation

It is recommended emitting events, for all the essential state variables that are possible to be changed during runtime.

Alleviation

The development team responded that they will fix this issue in the next version.

SIC-01 | Lack of input validation

Category	Severity	Location	Status
Logical Issue	● Informational	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol: 652, 675	ⓘ Acknowledged

Description

There is no validation to check whether `feeRate_` and `newRepoFeeRate_` are less than `PERCENTAGE_BASE`.

Recommendation

We advise the client to add a reasonable fee range for `newRepoFeeRate_` and `feeRate_`.

Alleviation

The development team responded that they will fix this issue in the next version.

SIC-02 | Boolean Equality Optimization

Category	Severity	Location	Status
Coding Style	● Informational	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol: 116	ⓘ Acknowledged

Description

Boolean constants can be used directly and do not need to be compared to true or false.

Recommendation

Consider removing the equality to the boolean constant as below:

```
        require(!initialized, "already initialized");
```

The code above is an example. Similar codes can also be modified.

Alleviation

The development team responded that they will fix this issue in the next version.

SIC-03 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	● Informational	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol : 115, 547, 639, 664, 669, 678, 746, 754, 767	ⓘ Acknowledged

Description

The following functions are declared as `public` and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

- `initialize()` in L115
- `remove()` in L547
- `_addMarket()` in L639
- `_removeMarket()` in L649
- `_setCurrency()` in L669
- `_withdrawFee()` in L678
- `_setPendingAdmin()` in L754
- `_acceptAdmin()` in L767

Recommendation

We advise that the functions' visibility specifiers are set to `external` and the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

Alleviation

No alleviation.

SIC-04 | Missing Input Validation

Category	Severity	Location	Status
Logical Issue	● Informational	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol : 187	ⓘ Acknowledged

Description

The given input is missing the sanity check for non-zero address in the aforementioned line.

Recommendation

We recommend adding the check for the passed-in values to prevent unexpected error as below: publishDecliningPrice():

```
187  require(icToken_ != address(0), "icToken_ address cannot be 0");
188  require(currency_ != address(0), "currency_ address cannot be 0");
```

Alleviation

The development team responded that they will fix this issue in the next version.

SIC-05 | Strengthen Transfer Security

Category	Severity	Location	Status
Logical Issue	● Minor	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol: 335, 376	ⓘ Acknowledged

Description

There are many transfer operations in functions `buyByAmount()` and `buyByUnits()`, adding a reentrant would be safer.

Recommendation

We advise the client to add a modifier as below:

```
bool private _status;
modifier nonReentrant() {
    require(!_status, 'reentrant call');
    _status = true;
    _;
   _status = false;
}
```

Alleviation

The development team responded that they will fix this issue in the next version.

SIC-06 | Missing Emit Events

Category	Severity	Location	Status
Logical Issue	● Informational	solv-v2-market/packages/solv-market/contracts/SolvICMarket.sol : 674	ⓘ Acknowledged

Description

The function that affects the status of sensitive variables should be able to emit events as notifications to customers.

- `_setRepoFeeRate()`

Recommendation

Consider adding events for sensitive actions, and emit them in the function.

Alleviation

The development team responded that they will fix this issue in the next version.

VNT-01 | Missing add `targetTokenId_` to `_slotTokens`

Category	Severity	Location	Status
Logical Issue	● Minor	solv-v2-voucher/packages/solv-vnft-core/contracts/VNFTCore.sol: 59~60	ⓘ Acknowledged

Description

When `targetTokenId_` does not exist, missing add the tokenId to the slot token list.

Recommendation

We advise the client to add the tokenId to `_slotTokens` as below:

```
        assets[tokenId_].transfer(assets[targetTokenId_], transferUnits_);
        if (! _slotTokens[slot_].contains(targetTokenId_)) {
            _slotTokens[slot_].add(targetTokenId_);
        }
        emit PartialTransfer(from_, to_, tokenId_, targetTokenId_, transferUnits_);
```

Alleviation

The development team responded that they will fix this issue in the next version.

VNT-02 | Optimization For Function `_burnUnits()`

Category	Severity	Location	Status
Logical Issue	● Informational	solv-v2-voucher/packages/solv-vnft-core/contracts/VNFTCore.sol : 142	ⓘ Acknowledged

Description

Is it necessary to burn the token and remove the related Information, when the `burnUnits` is equal to `assets[tokenId_]`?

Recommendation

We advise the client to change as below:

```
    function _burnUnits(uint256 tokenId_, uint256 burnUnits_) internal virtual returns
(uint256 balance) {
        if(assets[tokenId_].units == burnUnits_){
            _burn(uint256 tokenId_);
        }else{
            address owner = ownerOf(tokenId_);
            assets[tokenId_].burn(burnUnits_);
        }
        emit Burn(owner, tokenId_, burnUnits_);

        return assets[tokenId_].units;
    }
```

Alleviation

The development team responded that they will fix this issue in the next version.

VPC-01 | Boolean Equality Optimization

Category	Severity	Location	Status
Coding Style	● Informational	solv-v2-voucher/packages/solv-voucher/contracts/VestingPool.sol : 52	ⓘ Acknowledged

Description

Boolean constants can be used directly and do not need to be compared to true or false.

Recommendation

Consider removing the equality to the boolean constant as below:

```
        require(!initialized, "already initialized");
```

The code above is an example. Similar codes can also be modified.

Alleviation

The development team responded that they will fix this issue in the next version.

VPC-02 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	● Informational	solv-v2-voucher/packages/solv-voucher/contracts/VestingPool.sol: 51, 67, 425, 438	ⓘ Acknowledged

Description

The following functions are declared as `public` and are not invoked in any of the contracts contained within the project's scope. The functions that are never called internally within the contract should have external visibility.

- `initialize()` in L51
- `_setManager()` in L67
- `_setPendingAdmin()` in L425
- `_acceptAdmin()` in L438

Recommendation

We advise that the functions' visibility specifiers are set to `external` and the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

Alleviation

No alleviation.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE” AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER’S OR ANY OTHER PERSON’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK’S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER’S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED “AS IS” AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK’S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

CERTIK