Security Assessment

Galaxy

Jun 7th, 2021

Table of Contents

Summary

This report has been prepared for Galaxy smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract 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.

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 given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Galaxy
Description	StarNFT + SpaceStation
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/NFTGalaxy/GalaxyInternal/tree/master/contracts
Commits	<a95c497d544bedbdf25ca6f17ad0e2ac6ad23ccb>

Audit Summary

Delivery Date	Jun 07, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

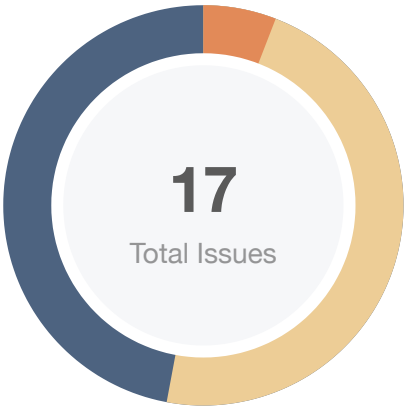
Vulnerability Summary

Total Issues	17
Critical	0
Major	1
Medium	0
Minor	8
Informational	8
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
SSS	SpaceStation/SpaceStation.sol	02e4f12bb19d5aad49e299384a8bb7de166d53a6c6f86e88974ffc3a64952d34
SNF	StarNFT/StarNFT.sol	7bfc62cc4f0c24adb1ae4e74aec5b978cbb2eccff5fd426d062fb0437d96bf30

Findings



Critical	**0** (0.00%)
Major	**1** (5.88%)
Medium	**0** (0.00%)
Minor	**8** (47.06%)
Informational	**8** (47.06%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
SNF-01	Missing zero address validation	Inconsistency	Minor	Resolved
SNF-02	Variable used out of scope	Gas Optimization	Minor	Resolved
SNF-03	Recommend double checking the data after initialization	Control Flow	Minor	Resolved
SNF-04	Recommend showing transparency to the community on the registered interfaces	**Mathematical Operations, Centralization / Privilege**	**Informational**	**Partially Resolved**
SNF-05	Redundant zero address check	Volatile Code	Informational	Resolved
SNF-06	Question about `galaxyCommunity`	Logical Issue	Informational	Partially Resolved
SNF-07	Privileged ownership	**Centralization / Privilege**	**Minor**	**Acknowledged**
SNF-08	Missing event emitting	Coding Style	Informational	Resolved
SNF-09	Typo	Volatile Code	Informational	Resolved
SNF-10	Proper usage of `require` statement in `isOwnerOf`	Control Flow	Minor	Resolved
SSS-01	Performing multiplication on the result of division	Mathematical Operations	Minor	Resolved
SSS-02	Missing event emitting	Coding Style	Informational	Resolved

ID	Title	Category	Severity	Status
SSS-03	Redundant code	Volatile Code	Informational	Resolved
SSS-04	Network fee charged multiple times	Control Flow	Major	Resolved
SSS-05	Privileged ownership	**Centralization / Privilege**	**Minor**	**Acknowledged**
SSS-06	Boolean equality	Volatile Code	Informational	Resolved
SSS-07	`_allowBlock` not set properly when early stake out allowed	Control Flow	Minor	Resolved

SNF-01 | Missing zero address validation

Category	Severity	Location	Status
Inconsistency	Minor	StarNFT/StarNFT.sol: 181, 199	Resolved

Description

The assigned address to the following functions should be verified as non zero value to prevent being mistakenly assigned as address(0) :

```
function transferOwner(address newOwner)
function safeTransferFrom(address to)
function safeBatchTransferFrom(address to)
```

Recommendation

We advise the client to check that the address is not zero by adding require statement in aformentioned functions.

Alleviation

The update has been applied to StarNFT contract.

https://github.com/NFTGalaxy/GalaxyInternal/commit/f36d66ad11fba48e960f529cd0137882a24ce4bb

SNF-02 | Variable used out of scope

Category	Severity	Location	Status
Gas Optimization	Minor	StarNFT/StarNFT.sol: 205	Resolved

Description

The variable `amount` is declared for each loop and then used out of scope.

Recommendation

Declare variable `amount` before scope of `for` loop (currently containing declaration).

Alleviation

The update has been applied to StarNFT contract.

https://github.com/NFTGalaxy/GalaxyInternal/commit/6024fecd210b17b71ad725ca3e9149c235eb9083

SNF-03 | Recommend double checking the data after initialization

Category	Severity	Location	Status
Control Flow	Minor	StarNFT/StarNFT.sol: 138	Resolved

Description

In initialize function, zero address check is needed to prevent losing ownership of StarNFT contract.

Recommendation

Add zero address check of `owner` and `galaxyCommunity` in initialize function.

Alleviation

The update has been applied to StarNFT contract.

https://github.com/NFTGalaxy/GalaxyInternal/commit/6024fecd210b17b71ad725ca3e9149c235eb9083

SNF-04 | Recommend showing transparency to the community on the registered interfaces

Category	Severity	Location	Status
Mathematical Operations, Centralization / Privilege	**Informational**	StarNFT/StarNFT.sol: 154 ~163	**Partially Resolved**

Description

In the current version, register interfaces are hardcoded in the initialize function.

Recommendation

We recommend showing details of how such interfaces are calculated in order to improve transparency to the community.

Alleviation

The update has been applied to StarNFT contract.

SNF-05 | Redundant zero address check

Category	Severity	Location	Status
Volatile Code	Informational	StarNFT/StarNFT.sol: 534	Resolved

Description

The zero address check has been made in L248 `mintSuper()` and it is redundant to check it again in `_mintSuper`.

Recommendation

Remove one of the checks in function `mintSuper` or `_mintSuper`

Alleviation

The update has been applied to StarNFT contract.

https://github.com/NFTGalaxy/GalaxyInternal/commit/b14363949e83f30225ba0ad66bf5d009d3b20597

SNF-06 | Question about `galaxyCommunity`

Category	Severity	Location	Status
Logical Issue	Informational	StarNFT/StarNFT.sol	Partially Resolved

Description

[Discussion] Are there any special reasons for separating the accessibilities of `owner` and `galaxyCommunity`? Who would be the `galaxyCommunity` after the protocol is released?

Alleviation

[Galaxy Team]: The governance module for galaxy community is still under discussion, we are only separating it out for future usage.

SNF-07 | Privileged ownership

Category	Severity	Location	Status
Centralization / Privilege	**Minor**	StarNFT/StarNFT.sol: (StarNFT)	**Acknowledged**

Description

The owner of contract `StarNFT` has the permission to:

1. transfer ownership of `StarNFT`,
2. add/remove minter, who then has the permission to Mint/Burn NFT tokens, which means whoever obtained access to the minter account would be able to tamper with the integrity of the token economics,
3. add/remove operator, who then has access to update/change any of their NFTs powah's immediately

without obtaining the consensus of the community.

Recommendation

Renounce ownership when it is the right timing, or gradually migrate to a timelock plus multisig governing procedure and let the community monitor in respect of transparency considerations.

We recommend that the team maintains a high level of transparency on such a transaction taking place.

Alleviation

The Galaxy team acknowledged this finding.

SNF-08 | Missing event emitting

Category	Severity	Location	Status
Coding Style	Informational	StarNFT/StarNFT.sol	Resolved

Description

In contract `StarNFT`, there are a bunch of functions can change state variables. However, these function do not emit event to pass the changes out of chain:

```
function transferGalaxyCommunity()
```

Recommendation

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

Alleviation

The update has been applied to contract StarNFT.sol

https://github.com/NFTGalaxy/GalaxyInternal/commit/a974a0642ba319019ff9ae9f9c671281520757eb

SNF-09 | Typo

Category	Severity	Location	Status
Volatile Code	Informational	StarNFT/StarNFT.sol: 314	Resolved

Description

Typo, The message should be "Operator already added".

Recommendation

Replace "Minter" with "Operator" in aformentioned message.

Alleviation

The update has been applied to StarNFT contract.

SNF-10 | Proper usage of `require` statement in `isOwnerOf`

Category	Severity	Location	Status
Control Flow	Minor	StarNFT/StarNFT.sol: 412	Resolved

Description

In L467, `balanceofBatch` called `isOwnerOf` in the for loop, if one of `accounts[i]` is `address(0)`,then the require check in `isOwnerOf` will revert and would revert the entire loop in `balanceOfBatch` .

Recommendation

We recommend using the following code:

```
function isOwnerOf(address account, uint256 id) public view override returns (bool) {
      if (account == address(0)){
        return false;
    } else {
        return _starBelongTo[id] == account;
    }
}
```

Alleviation

The update has been applied to StarNFT contract.

SSS-01 | Performing multiplication on the result of division

Category	Severity	Location	Status
Mathematical Operations	Minor	SpaceStation/SpaceStation.sol: 414~419, 446~451, 524~530	Resolved

Description

Solidity integer division might truncate. As a result, performing multiplication before division can sometimes avoid loss of precision.

Recommendation

Consider ordering multiplication before division.

Alleviation

The update has been applied to SpaceStation contract.

https://github.com/NFTGalaxy/GalaxyInternal/commit/68738bdbf78c0c45fb707e7970c54b704026770e

SSS-02 | Missing event emitting

Category	Severity	Location	Status
Coding Style	Informational	SpaceStation/SpaceStation.sol	Resolved

Description

In contract `SpaceStation`, there are a bunch of functions can change state variables. However, these function do not emit event to pass the changes out of chain.

```
function activateCampaign()

function expireCampaign()

function activateStakeCampaign()
```

Recommendation

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

Alleviation

The update has been applied to contract SpaceStation.sol

https://github.com/NFTGalaxy/GalaxyInternal/commit/a974a0642ba319019ff9ae9f9c671281520757eb

SSS-03 | Redundant code

Category	Severity	Location	Status
Volatile Code	Informational	SpaceStation/SpaceStation.sol: 533~535	Resolved

Description

The `else` branch in L533 is redundant if we put the `require` check in L522.

Recommendation

Here is the example code:

```
if (block.number < _mintBlock + lockBlockNum) {
            require(campaignStakeConfigs[_cid].isEarlyStakeOutAllowed, "Early stake out
not allowed")
            // calc fine if allow early stake out
            uint256 _fine = (_mintBlock + lockBlockNum)
            .sub(block.number)
            .mul(10000)
            .div(lockBlockNum)
            .mul(campaignStakeConfigs[_cid].earlyStakeOutFine)
            .div(10000);
            require(msg.value >= campaignFeeConfigs[_cid]
[Operation.StakeOut].networkFee.add(_fine), "Insufficient fine");
            // Fine should be adding to treasury
            galaxyTreasuryNetwork = galaxyTreasuryNetwork.add(_fine);
        }
...
```

Alleviation

The update has been applied to contract SpaceStation.sol

https://github.com/NFTGalaxy/GalaxyInternal/commit/69e3444f0752a6d721d985cd053adb5fa15e8409

SSS-04 | Network fee charged multiple times

Category	Severity	Location	Status
Control Flow	Major	SpaceStation/SpaceStation.sol: 231~233	Resolved

Description

In function `stakeOutQuasar()` , network fee is charged by calling `_payFine()`, however it is charged a second time in `_payFees()`.

L499 in `_payFees()`

```
 if (feeConf.networkFee > 0) {
            require(msg.value >= feeConf.networkFee, "Invalid msg.value sent for
networkFee");
            galaxyTreasuryNetwork = galaxyTreasuryNetwork.add(msg.value);
        }
```

L530 in `_payFine()`

```
require(msg.value >= campaignFeeConfigs[_cid][Operation.StakeOut].networkFee.add(_fine),
"Insufficient fine");
                // Fine should be adding to treasury
                galaxyTreasuryNetwork = galaxyTreasuryNetwork.add(_fine);
```

Besides, the charged network fee in `_payFine()` is not added into `galaxyTreasuryNetwork`.

Recommendation

Don't charge network fee in `_payFine`. If the team want to charge network fee in both `_payFees()` and `_payFine()`, please add the charged network fee into `galaxyTreasuryNetwork` in L532.

Alleviation

The update has been applied to contract SpaceStation.sol

https://github.com/NFTGalaxy/GalaxyInternal/commit/aec78e8378bfe69a224ba2d8207f32ed6d4e7891

SSS-05 | Privileged ownership

Category	Severity	Location	Status
Centralization / Privilege	**Minor**	SpaceStation/SpaceStation.sol	**Acknowledged**

Description

There are two privileged roles in `SpaceStation`

The manager of contract `SpaceStation` has the permission to:

1. activate Campaign and set fee parameters,
2. set Campaign as expired and delete it,
3. pause the contract,
4. add/remove validated StarNFT address

without obtaining the consensus of the community.

The Treasurymanager of contract `SpaceStation` has the permission to:

1. Withdraw eth/erc20 tokens from `galaxyTreasury` and transfer it to contract manager

without obtaining the consensus of the community.

Recommendation

Gradually migrate the formentioned roles to a timelock plus multisig governing procedure and let the community monitor in respect of transparency considerations.

Alleviation

The Galaxy team acknowledged this finding.

SSS-06 | Boolean equality

Category	Severity	Location	Status
Volatile Code	Informational	SpaceStation/SpaceStation.sol: 51	Resolved

Description

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

Recommendation

Remove the equality to the boolean constant.

```
require(_starNFTs[_starNFTAddress], "Invalid Star NFT contract address");
```

Alleviation

The update has been applied to contract SpaceStation.sol

SSS-07 | `_allowBlock` not set properly when early stake out allowed

Category	Severity	Location	Status
Control Flow	Minor	SpaceStation/SpaceStation.sol: 407, 439	Resolved

Description

In the case `campaignStakeConfigs[_cid].isEarlyStakeOutAllowed = true`, the variable `_allowBlock` is still equal to `_noFineBlock`. Literally, `_allowBlock` should be `_createBlock` since early stake out is allowed although early stake fine is charged.

Recommendation

Set the `_allowBlock` as `_createBlock` for the last case of `if-else` branch in L412 and L444.

```
        _allowBlock = campaignStakeConfigs[_cid].lockBlockNum + _createBlock;
        if (!campaignStakeConfigs[_cid].isEarlyStakeOutAllowed) {
            // not allow early stakeout
            return (false, _allowBlock, _requireBurn, 0, 0);
        }
        _allowStakeOut = true;
        _allowBlock = _createBlock;
        _noFineBlock = _createBlock + campaignStakeConfigs[_cid].lockBlockNum;
...
```

Alleviation

The update has been applied to contract SpaceStation.sol

https://github.com/NFTGalaxy/GalaxyInternal/commit/b44d0bde98086b7032e39ead0c05868acb59f200

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.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

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.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

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.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

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 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 without CertiK's prior written consent.

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.

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.