Security Assessment

Intelligent Mining

Aug 31st, 2021

Table of Contents

Summary

This report has been prepared for Intelligent Mining to discover issues and vulnerabilities in the source code of the Intelligent Mining 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.

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	Intelligent Mining
Description	DeFi + Staking
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/IM-Intelligent-Mining/contracts/tree/master/ERC721-timelock/contracts https://github.com/IM-Intelligent-Mining/contracts/tree/master/im-plus/contracts
Commit	375ee2eb634aa80b03dff54ef12cf7247463d4f3

Audit Summary

Delivery Date	Aug 31, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

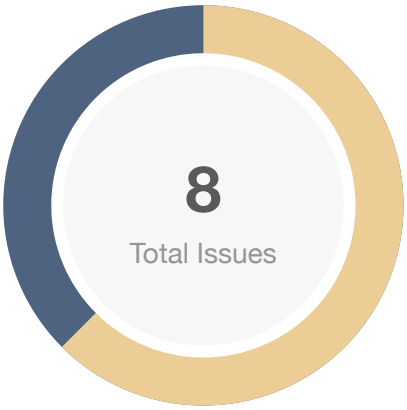
Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	0	0	0	0	0	0
Medium	0	0	0	0	0	0
Minor	5	0	0	2	0	3
Informational	3	0	0	2	0	1
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
AER	ERC721-timelock/contracts/Address.sol	eb410a76a8fdebb94bda037322b22223c58c8afd4d4f8c26626f2822386ef994
CER	ERC721-timelock/contracts/Context.sol	9a3d1e5be0f0ace13e2d9aa1d0a1c3a6574983983ad5de94fc412f878bf7fe89
ERE	ERC721-timelock/contracts/ERC165.sol	1f45e819a9556fe1d72270ae2fd57a941a2506e965df9fd7e26566aa39cf4460
ERR	ERC721-timelock/contracts/ERC721.sol	5c19e9451879fe0131c28785438bdfe99a9dcd879e1fb140f24356fa605e440e
EME	ERC721-timelock/contracts/EnumerableMap.sol	26c7ec2df617e9420a3782d911dc6c339e83b02eac442de4c3c4bbbd18fe3273
ESE	ERC721-timelock/contracts/EnumerableSet.sol	c8b73a000476872a00f6153d66be31a4a99b7565068f05336129748bfad704ea
IER	ERC721-timelock/contracts/IERC165.sol	24d63fd063d0d9e954ce1a039404b4c01d2141f787143bbd3d5090a0220a2bcc
IEC	ERC721-timelock/contracts/IERC721.sol	1802b20515694649f4e98bca15248b1caefcb5bf454ac50f99b8bef353fa3833
IEE	ERC721-timelock/contracts/IERC721Enumerable.sol	da6fa0593fd96281d88df725727540d0c61551ed756a31a2ef6e1e8ccfbbe59d
IEM	ERC721-timelock/contracts/IERC721Metadata.sol	17a75a430e00aa592ec076cecb7c1fee37b4b21c10cec9b84f57faac13fb3cb5
IEK	ERC721-timelock/contracts/IERC721Receiver.sol	7e3d89b564e70918bc4e71e8346271f90dc3359d65b542baf24ce4de4e73d0a8
SME	ERC721-timelock/contracts/SafeMath.sol	c6d17ae2340573ed1eb5930fbd8f62f46246c443ed62412edbf8803305769331
SER	ERC721-timelock/contracts/Strings.sol	c3c3a9561de5e096929024e8a5476d6982dfa5c85065624fa94c358848c5285d
TTE	ERC721-timelock/contracts/TokenTimelock.sol	86f4befde913be1534efd144752a561a491c9ce204e32dfa41ac2153266b1b4f
CCK	im-plus/contracts/Context.sol	71451b3a77f4cc8386eebc9fb02a798ffad77f6a236c5f0fa58a00eb343f6768

ID	File	SHA256 Checksum
ERP	im-plus/contracts/ERC20.sol	f55aae6922461a9ae61aa762542aa493b67e5e8e4d2bb88accd80b3e4618a7fd
IEP	im-plus/contracts/IERC20.sol	0573c2961569aa4906845d0cd428b5b7394956170054ceeaa8f8af96cd44875c
IRC	im-plus/contracts/IERC20Metadata.sol	666d8664b70860d06f481cd4f1e2aa3a8d54582f007c9d4232a8c362ce042b5f
IPT	im-plus/contracts/ImPlusToken.sol	dd2e7f700d60b861979b305c260a4589e306aa3044cf663d13cd227d575f2199
OCK	im-plus/contracts/Ownable.sol	3c80921ce9cbc5099e446ad07ec43bbe55d6e8170e4addd844a5177ffaa13901
SMC	im-plus/contracts/SafeMath.sol	c6d17ae2340573ed1eb5930fbd8f62f46246c443ed62412edbf8803305769331

Findings



Critical	**0** (0.00%)
Major	**0** (0.00%)
Medium	**0** (0.00%)
Minor	**5** (62.50%)
Informational	**3** (37.50%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
ERP-01	Typo in the comments	Language Specific	Informational	Resolved
ERR-01	Third Party Dependencies	Volatile Code	Minor	Acknowledged
ERR-02	Function Return Value Ignored	Volatile Code	Informational	Acknowledged
IPT-01	Potential Reentrancy risks	Volatile Code	Minor	Resolved
IPT-02	Centralization Risk	**Centralization / Privilege**	**Minor**	Acknowledged
IPT-03	Potential Overflow	Mathematical Operations	Minor	Resolved
IPT-04	Function Return Value Ignored	Volatile Code	Informational	Acknowledged
TTE-01	Missing Zero Address Validation	Volatile Code	Minor	Resolved

ERP-01 | Typo in the comments

Category	Severity	Location	Status
Language Specific	Informational	im-plus/contracts/ERC20.sol: 232	Resolved

Description

"`to` cannot be the zero address". Written in the comments of function `_mint(address account, uint256 amount)`. While no parameter in `_mint()` named `to`.

Recommendation

We recommend revising the parameter name in the comment.

Alleviation

The team heeded our advice and fixed the issue in commit f887322a72f3f57e079176b01ab3978d79d7e8cf.

ERR-01 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Minor	ERC721-timelock/contracts/ERC721.sol: 441~447	Acknowledged

Description

The contract is serving as the underlying entity to interact with a third party in the code below:

```
bytes memory returndata = to.functionCall(abi.encodeWithSelector(
    IERC721Receiver(to).onERC721Received.selector,
    _msgSender(),
    from,
    tokenId,
    _data
), "ERC721: transfer to non ERC721Receiver implementer");
```

The scope of the audit treats 3rd party entities as black boxes and assume their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Recommendation

We understand that the business logic of function `_checkOnERC721Received()` requires interaction with a third party. We recommend the team constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

The team acknowledged the finding and decided to retain the code unchanged.

ERR-02 | Function Return Value Ignored

Category	Severity	Location	Status
Volatile Code	Informational	ERC721-timelock/contracts/ERC721.sol: 339, 341, 369, 371, 396, 397, 399	Acknowledged

Description

The following issues found:

```
1  ERC721._mint(address,uint256) ignores return value by _holderTokens[to].add(tokenId)
2  ERC721._mint(address,uint256) ignores return value by _tokenOwners.set(tokenId,to)
3  ERC721._burn(uint256) ignores return value by _holderTokens[owner].remove(tokenId)
4  ERC721._burn(uint256) ignores return value by _tokenOwners.remove(tokenId)
5  ERC721._transfer(address,address,uint256) ignores return value by
_holderTokens[from].remove(tokenId)
6  ERC721._transfer(address,address,uint256) ignores return value by
_holderTokens[to].add(tokenId)
7  ERC721._transfer(address,address,uint256) ignores return value by
_tokenOwners.set(tokenId,to)
8  ImPlusToken.emergencyWithdrawn() ignores return value by
im.transfer(owner(),lockFunds)
```

Recommendation

We recommend developers to handle all return values of aforementioned function.

Alleviation

The team acknowledged the finding and decided to retain the code unchanged.

IPT-01 | Potential Reentrancy risks

Category	Severity	Location	Status
Volatile Code	Minor	im-plus/contracts/ImPlusToken.sol: 46~58	Resolved

Description

Function `stake()` is risky to reentrancy attacks. Variables `startLockTime[_msgSender()]`, `_stakeBalances[_msgSender()]` and `totalStaking` will be changed after `im.transferFrom()` is called. Since the implementation of the external function and the address behind the interface are unknown, reentrancy is possible to take place.

Recommendation

We recommend applying the Checks-Effects-Interactions Pattern to avoid the risk of calling unknown contracts.

Alleviation

The team heeded our advice and fixed the issue in commit f887322a72f3f57e079176b01ab3978d79d7e8cf.

IPT-02 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Minor	im-plus/contracts/ImPlusToken.sol: 77~83	Acknowledged

Description

```
function emergencyWithdrawn() external onlyOwner {
    uint256 contractTokenHold = im.balanceOf(address(this));
    uint256 lockFunds = contractTokenHold - totalStaking;
    require(lockFunds > 0, "No lock funds");

    im.transfer(owner(), lockFunds);
}
```

The owner has the authority to withdraw tokens that are accidentally transferred into the contract with the function `emergencyWithdrawn()`. However, we also need to point out that the owner is NOT capable to withdraw tokens staked in the pool with the function.

Recommendation

We recommend the client carefully manage the `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 is some feasible suggestions that would also mitigate the potential risk at the different level in term 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

Since the IM team can only pull out tokens that were sent to the contract by mistake, they decided to retain the code unchanged.

IPT-03 | Potential Overflow

Category	Severity	Location	Status
Mathematical Operations	Minor	im-plus/contracts/ImPlusToken.sol: 54~55, 104	Resolved

Description

Overflows may take place in the equations below:

```
startLockTime[_msgSender()] = block.timestamp;
_stakeBalances[_msgSender()] += _amount;
rewards[_msgSender()] += amount;
```

Recommendation

We recommend utilizing the function `add()` in the library `SafeMath` for these equations.

IPT-04 | Function Return Value Ignored

Category	Severity	Location	Status
Volatile Code	Informational	im-plus/contracts/ImPlusToken.sol: 82	Acknowledged

Description

The following issues found:

```
1 ERC721._mint(address,uint256) ignores return value by _holderTokens[to].add(tokenId)
2 ERC721._mint(address,uint256) ignores return value by _tokenOwners.set(tokenId,to)
3 ERC721._burn(uint256) ignores return value by _holderTokens[owner].remove(tokenId)
4 ERC721._burn(uint256) ignores return value by _tokenOwners.remove(tokenId)
5 ERC721._transfer(address,address,uint256) ignores return value by
_holderTokens[from].remove(tokenId)
6 ERC721._transfer(address,address,uint256) ignores return value by
_holderTokens[to].add(tokenId)
7 ERC721._transfer(address,address,uint256) ignores return value by
_tokenOwners.set(tokenId,to)
8 ImPlusToken.emergencyWithdrawn() ignores return value by
im.transfer(owner(),lockFunds)
```

Recommendation

We recommend developers to handle all return values of aforementioned function.

Alleviation

The team acknowledged the finding and decided to retain the code unchanged.

TTE-01 | Missing Zero Address Validation

Category	Severity	Location	Status
Volatile Code	Minor	ERC721-timelock/contracts/TokenTimelock.sol: 31	Resolved

Description

The assigned value to `_beneficiary` should be verified as non-zero value to prevent being mistakenly assigned as address(0) in `constructor()` function. Violation of this may cause losing ownership.

```
- _beneficiary = beneficiary_ (TokenTimelock.sol#31)
```

Recommendation

We recommend checking that the address is not zero by adding checks in function.

Alleviation

The team heeded our advice and fixed the issue in commit f887322a72f3f57e079176b01ab3978d79d7e8cf.

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.

Mathematical Operations

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

Volatile Code

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

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

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.