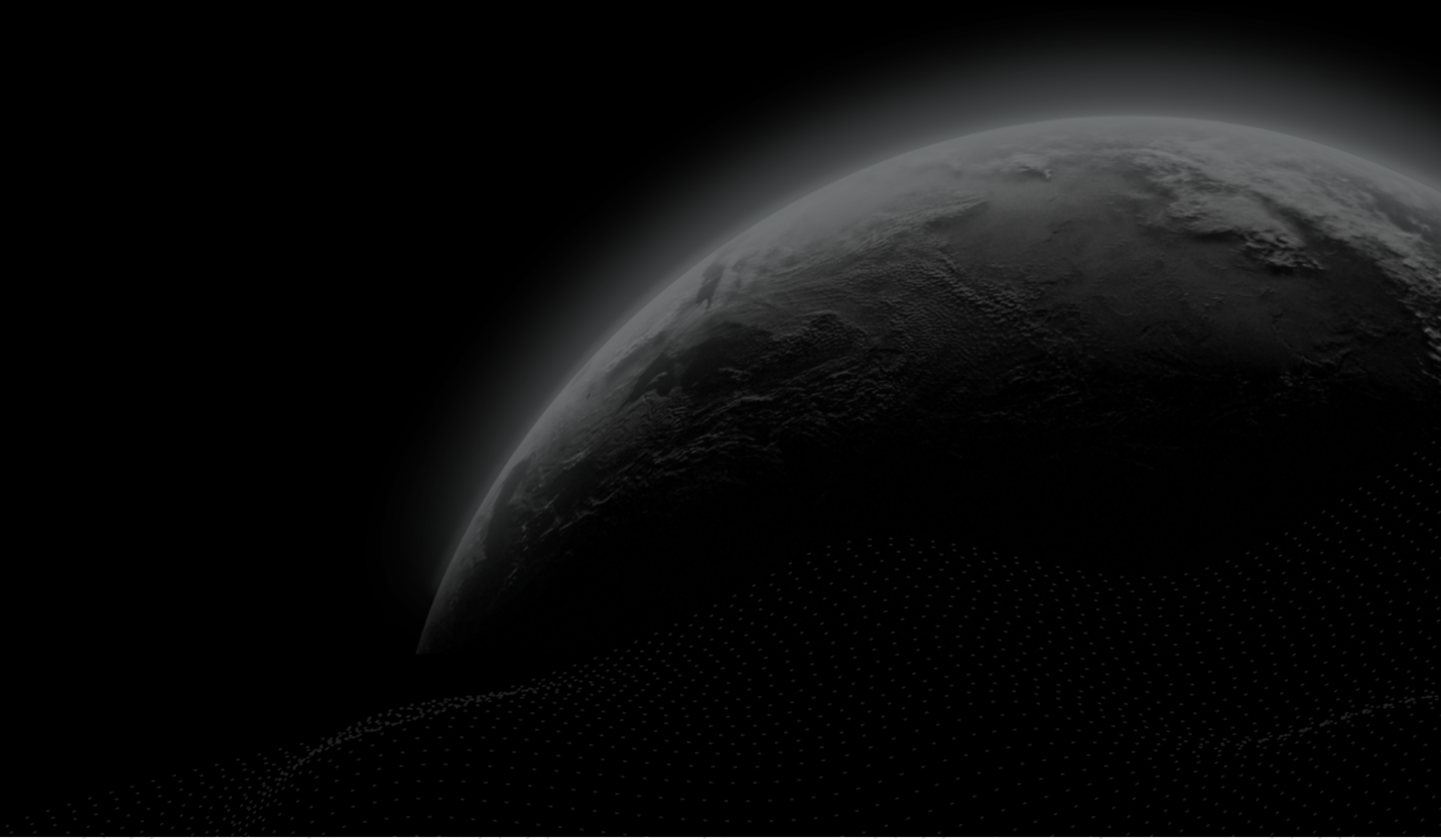
CERTIK

Security Assessment

Draft (Internal Use Only)

BMX Token

CertiK Verified on Aug 30th, 2022



CertiK Verified on Aug 30th, 2022

BMX Token

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES
ERC-20

ECOSYSTEM
Ethereum

METHODS
Manual Review, Static Analysis

LANGUAGE
Solidity

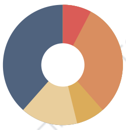
TIMELINE
Delivered on 08/30/2022

KEY COMPONENTS
N/A

CODEBASE
https://github.com/bitmartexchange/bitmart-smart-contract
...View All

COMMITS
389dd8b08ca7c2e6d74f5f9da773004a98cd140f
f651d70ab8d86978bbb40df848d56da0585ed1fd
49bfb473ea3130292df936b698ecd1e31f6e73fa
...View All

Vulnerability Summary



13	2	3	0	8	0	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

🟥 1	Critical	1 Mitigated	Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟧 4	Major	1 Resolved, 2 Mitigated, 1 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟨 1	Medium	1 Acknowledged	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨 2	Minor	1 Resolved, 1 Acknowledged	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
🟦 5	Informational	5 Acknowledged	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | BMX TOKEN

Summary

Executive Summary

Vulnerability Summary

Codebase

Audit Scope

Approach & Methods

Findings

BMX-01 : Centralization Risks in BMX.sol

BMX-02 : Potential Risk On `approve()`/`transferFrom()` Methods

BMX-03 : Incorrect ERC-20 Interface

BMX-04 : Usage of `transfer()` for sending Ether

TIM-01 : Excessive owner privileges

TIM-02 : Centralization Risks in Timelock.sol

TIM-03 : Inappropriate Access Control

TIM-04 : Missing Zero Address Validation

389-02 : Solidity Version Not Recommended

BMX-05 : New Syntax for Constructor

BMX-06 : New Syntax for Fallback Function

BMX-07 : New Syntax for Emitting Events

BMX-08 : Missing Error Messages

Optimizations

389-01 : Unnecessary Use of SafeMath

Appendix

Disclaimer

CODEBASE | BMX TOKEN

Repository

https://github.com/bitmartexchange/bitmart-smart-contract

Commit

389dd8b08ca7c2e6d74f5f9da773004a98cd140f f651d70ab8d86978bbb40df848d56da0585ed1fd
49bfb473ea3130292df936b698ecd1e31f6e73fa

AUDIT SCOPE | BMX TOKEN

3 files audited ● 3 files with Acknowledged findings

ID	File	SHA256 Checksum
● BMX	BMX.sol	48f1696ef1ea35571bf21720d9548b21f9dab6760beb7277593c30e261b69df7
● SMB	SafeMath.sol	040cd64e2ecd619d78e184630228191656a0abd0e8de867610dd89f1610dfe32
● TIM	Timelock.sol	fc6020e02729543164ed3128b5abe128aacf481f44a98ce67a3d164cdc3fd58f

APPROACH & METHODS | BMX TOKEN

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

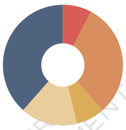
The auditing process pays special attention to the following considerations:

- 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:

- Testing the smart contracts against both common and uncommon attack vectors;
- 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.

FINDINGS | BMX TOKEN



13	1	4	1	2	5
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for BMX Token. Through this audit, we have uncovered 13 issues ranging from different severity levels. Utilizing Static Analysis techniques to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
BMX-01	**Centralization Risks In BMX.Sol**	**Centralization / Privilege**	**Major**	● **Mitigated**
BMX-02	Potential Risk On `approve()` / `transferFrom()` Methods	Volatile Code	Major	● Acknowledged
BMX-03	Incorrect ERC-20 Interface	Language Specific	Medium	● Acknowledged
BMX-04	Usage Of `transfer()` For Sending Ether	Volatile Code	Minor	● Acknowledged
TIM-01	**Excessive Owner Privileges**	**Centralization / Privilege**	**Critical**	● **Mitigated**
TIM-02	**Centralization Risks In Timelock.Sol**	**Centralization / Privilege**	**Major**	● **Mitigated**
TIM-03	Inappropriate Access Control	Logical Issue	Major	● Resolved
TIM-04	Missing Zero Address Validation	Volatile Code	Minor	● Resolved
389-02	Solidity Version Not Recommended	Language Specific	Informational	● Acknowledged
BMX-05	New Syntax For Constructor	Language Specific	Informational	● Acknowledged
BMX-06	New Syntax For Fallback Function	Language Specific	Informational	● Acknowledged

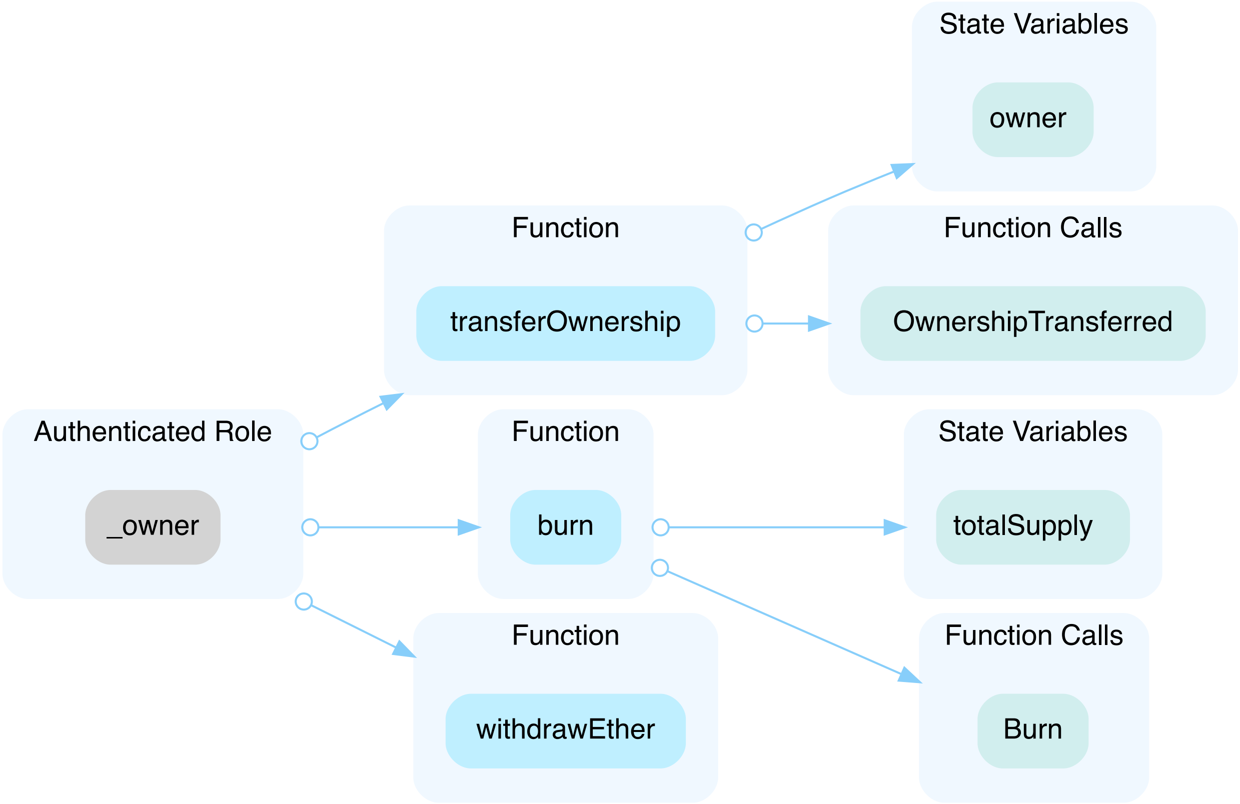
ID	Title	Category	Severity	Status
BMX-07	New Syntax For Emitting Events	Language Specific	Informational	● Acknowledged
BMX-08	Missing Error Messages	Coding Style	Informational	● Acknowledged

BMX-01 | CENTRALIZATION RISKS IN BMX.SOL

Category	Severity	Location	Status
Centralization / Privilege	● Major	BMX.sol: 98, 138, 169	● Mitigated

Description

In the contract `BMC` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and drain all eth out of this contract simply by calling the withdrawEther() function.



Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol 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 a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

`[CertiK]` : The `BMX` token deployment is at the address 0x986ee2b944c42d017f52af21c4c69b84dbea35d8 and the address of the `owner` role is 0x5B7b9C0a51Eaf0324815cF8Fb6ffD84Ab2900069, which is a `Timelock` contract.

The `admin` of the `Timelock` deployment is 0x53e5FE23807489F5FE1e07f11A464f04A79F9e82, which is a Gnosis Safe deployment.

Any transaction requires the confirmation of 2 out of 3 following signers:

- 0x5ca2fa4a38edcb9e874c1581be0a06aa1a58cb08

- 0x7623641bc08d716f08a6b747a45290b1a68419d8
- 0xD91A8cBF3EAB3DC0e322364F7536B766C843fC64

The team also published detailed decentralization efforts in the URL https://www.bitmart.com/bmx/en

BMX-02 | POTENTIAL RISK ON `approve()` / `transferFrom()` METHODS

Category	Severity	Location	Status
Volatile Code	● Major	BMX.sol: 117, 124	● Acknowledged

Description

The `approve` function could be used in a Front-Running attack that allows a spender to transfer more tokens than the owner of the tokens ever wanted to allow the spender to transfer.

Here is a possible attack scenario:

Alice allows Bob to transfer N of Alice's tokens (N>0) by calling approve method on Token smart contract passing Bob's address and N as method arguments After some time, Alice decides to change from N to M (M>0) the number of Alice's tokens Bob is allowed to transfer, so she calls approve method again, this time passing Bob's address and M as method arguments Bob notices Alice's second transaction before it was mined and quickly sends another transaction that calls ·transferFrom· method to transfer N Alice's tokens somewhere If Bob's transaction will be executed before Alice's transaction, then Bob will successfully transfer N Alice's tokens and will gain the ability to transfer another M tokens Before Alice noticed that something went wrong, Bob calls ·transferFrom· method again, this time to transfer M Alice's tokens.

So, Alice's attempt to change Bob's allowance from N to M (N>0 and M>0) made it possible for Bob to transfer N+M of Alice's tokens, while Alice never wanted to allow so many of her tokens to be transferred by Bob.

Recommendation

We advise the client to use functions like `increaseAllowance()` and `decreaseAllowance()` from the `ERC20.sol` contract from OpenZeppelin.

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

BMX-03 | INCORRECT ERC-20 INTERFACE

Category	Severity	Location	Status
Language Specific	● Medium	BMX.sol: 105	● Acknowledged

Description

Incorrect return values for ERC-20 functions. A contract compiled with Solidity > 0.4.22 interacting with these functions will fail to execute them, as the return value is missing.

BMC (BMX.sol#48-176) has incorrect ERC20 function interface:BMC.transfer(address,uint256) (BMX.sol#105-114)

```
105     function transfer(address _to, uint256 _value) public {
```

Recommendation

We recommend setting the appropriate return values and types for the defined ERC-20 functions.

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

BMX-04 | USAGE OF `transfer()` FOR SENDING ETHER

Category	Severity	Location	Status
Volatile Code	● Minor	BMX.sol: 170~171	● Acknowledged

Description

After EIP-1884 was included in the Istanbul hard fork, it is not recommended to use `.transfer()` or `.send()` for transferring ether as these functions have a hard-coded value for gas costs making them obsolete as they are forwarding a fixed amount of gas, specifically `2300`. This can cause issues in case the linked statements are meant to be able to transfer funds to other contracts instead of EOAs.

Recommendation

We advise that the linked `.transfer()` call is substituted with the utilization of the `sendValue()` function from the `Address.sol` implementation of OpenZeppelin either by directly importing the library or copying the linked code.

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

TIM-01 | EXCESSIVE OWNER PRIVILEGES

Category	Severity	Location	Status
Centralization / Privilege	● Critical	Timelock.sol: 82, 96, 100	● Mitigated

Description

```
bytes memory callData1 = abi.encodePacked(bytes4(keccak256(bytes(signature))),
data);
bytes memory callData2 = abi.encodeWithSignature(signature, data);
```

`callData1` is equal to `callData2`. This means the `admin` of this contract can execute any function in any contract.

Recommendation

Excessive owner privileges We advise the client to carefully manage the `admin` 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 via smart-contract-based accounts with enhanced security practices, e.g. Multisignature wallets. Here are some feasible solutions that would also mitigate the potential risk:

- 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

`[CertiK]`: The `BMX` token deployment is at the address 0x986ee2b944c42d017f52af21c4c69b84dbea35d8 and the address of the `owner` role is 0x5B7b9C0a51Eaf0324815cF8Fb6ffD84Ab2900069, which is a `Timelock` contract.

The `admin` of the `Timelock` deployment is 0x53e5FE23807489F5FE1e07f11A464f04A79F9e82, which is a Gnosis Safe deployment.

Any transaction requires the confirmation of 2 out of 3 following signers:

- 0x5ca2fa4a38edcb9e874c1581be0a06aa1a58cb08
- 0x7623641bc08d716f08a6b747a45290b1a68419d8
- 0xD91A8cBF3EAB3DC0e322364F7536B766C843fC64

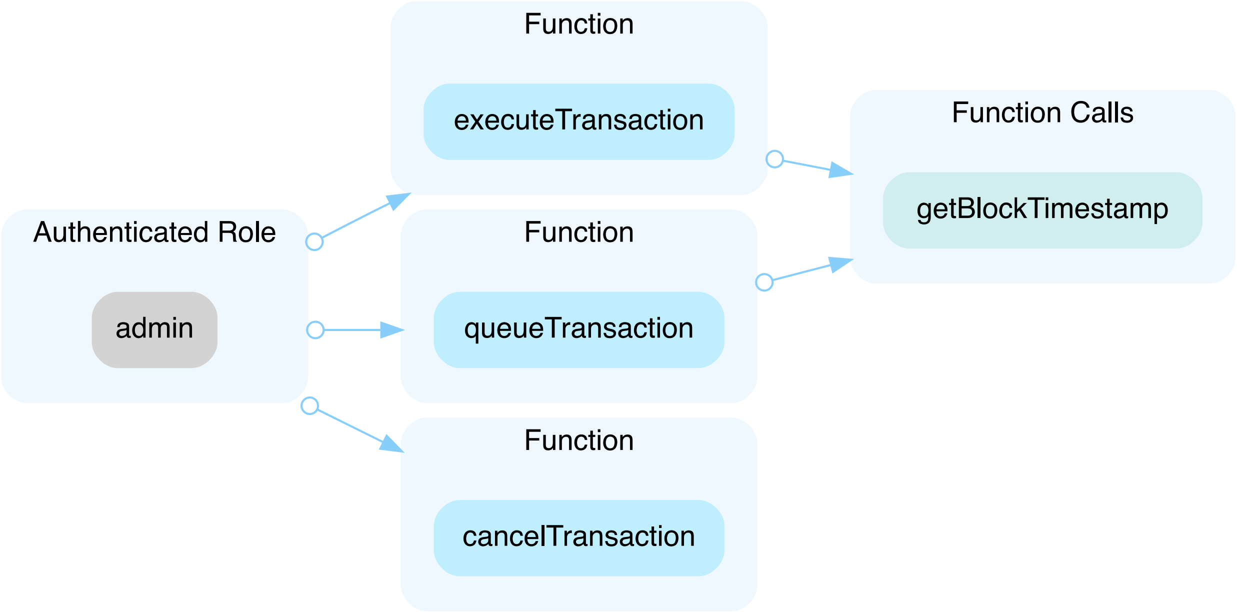
The team also published detailed decentralization efforts in the URL https://www.bitmart.com/bmx/en

TIM-02 | CENTRALIZATION RISKS IN TIMELOCK.SOL

Category	Severity	Location	Status
Centralization / Privilege	● Major	Timelock.sol: 46, 61, 72, 81	● Mitigated

Description

In the contract `Timelock` the role `admin` has authority over the functions shown in the diagram below. Any compromise to the `admin` account may allow the hacker to take advantage of this authority and repeatedly add transactions by calling the queueTransaction() function and next call the executeTransaction() function to drain funds out of this contract.



Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol 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 a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

`[CertiK]` : The `BMX` token deployment is at the address 0x986ee2b944c42d017f52af21c4c69b84dbea35d8 and the address of the `owner` role is 0x5B7b9C0a51Eaf0324815cF8Fb6ffD84Ab2900069, which is a `Timelock` contract.

The `admin` of the `Timelock` deployment is 0x53e5FE23807489F5FE1e07f11A464f04A79F9e82, which is a Gnosis Safe deployment.

Any transaction requires the confirmation of 2 out of 3 following signers:

- 0x5ca2fa4a38edcb9e874c1581be0a06aa1a58cb08
- 0x7623641bc08d716f08a6b747a45290b1a68419d8
- 0xD91A8cBF3EAB3DC0e322364F7536B766C843fC64

The team also published detailed decentralization efforts in the URL https://www.bitmart.com/bmx/en

TIM-03 | INAPPROPRIATE ACCESS CONTROL

Category	Severity	Location	Status
Logical Issue	● Major	Timelock.sol: 38, 55~56	● Resolved

Description

The linked require statements mean that only this contract can call them but there is no function in this contract actually call these two functions. Hence, function setDelay(uint) and setPendingAdmin(address) are unable.

Recommendation

We recommend double checking the codebase.

Alleviation

`[BMX Token]` : The calls of `setDelay(uint)` and `setPendingAdmin(address)` methods need to be executed by calling `queuetransaction()` and `executetransaction()` . This is because all operations in timelock need to comply with the locking rules.

TIM-04 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	● Minor	Timelock.sol: 31, 56, 100	● Resolved

Description

Addresses should be checked before assignment or external call to make sure they are not zero addresses.

```
31          admin = admin_;
```

- `admin_` is not zero-checked before being used.

```
56          pendingAdmin = pendingAdmin_;
```

- `pendingAdmin_` is not zero-checked before being used.

```
 100          (bool success, bytes memory returnData) = target.call{value: value}
(callData);
```

- `target` is not zero-checked before being used.

Recommendation

We advise adding a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

`[BMX Token]` : Issue resolved. Changes have been reflected in the commit hash 49bfb473ea3130292df936b698ecd1e31f6e73fa

389-02 | SOLIDITY VERSION NOT RECOMMENDED

Category	Severity	Location	Status
Language Specific	● Informational	BMX.sol: 1; SafeMath.sol: 2; Timelock.sol: 2	● Acknowledged

Description

Solidity frequently releases new compiler versions. Using an old version prevents access to new Solidity security features. Also, recent versions may be too early to be trusted.

solc-0.4.26 is not recommended for deployment

solc-0.8.15 is not recommended for deployment

Pragma version^0.4.18 (BMX.sol#1) allows old versions

```
1  pragma solidity ^0.4.18;
```

Pragma version^0.8.10 (SafeMath.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7

```
2  pragma solidity ^0.8.10;
```

Pragma version^0.8.10 (Timelock.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7

```
2  pragma solidity ^0.8.10;
```

Recommendation

We recommend deploying with any of the following Solidity versions:

- 0.5.16 - 0.5.17
- 0.6.11 - 0.6.12
- 0.7.5 - 0.7.6

Use a simple pragma version that allows any of these versions. Also, consider using the latest version of Solidity for testing.

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

BMX-05 | NEW SYNTAX FOR CONSTRUCTOR

Category	Severity	Location	Status
Language Specific	● Informational	BMX.sol: 77~84	● Acknowledged

Description

Since Solidity v0.4.23, constructors are now specified using the constructor keyword. Using the name of a contract as its constructor is now deprecated.

Recommendation

We recommend applying the new follwing new syntax:

```
    constructor( uint256 initialSupply, uint8 decimalUnits) public {
        balanceOf[msg.sender] = initialSupply; // Give the creator all initial
tokens
        totalSupply = initialSupply; // Update total supply
        name = "BitMartToken";    // Set the name for display purposes
        symbol = "BMC";     // Set the symbol for display purposes
        decimals = decimalUnits;  // Amount of decimals for display purposes
        owner = msg.sender;
    }
```

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

BMX-06 | NEW SYNTAX FOR FALLBACK FUNCTION

Category	Severity	Location	Status
Language Specific	● Informational	BMX.sol: 174~175	● Acknowledged

Description

The fallback function now has a different syntax, declared using fallback() external [payable] {...} (without the function keyword).

Recommendation

We recommend applying the follwing syntax.

```
fallback() external payable{
    // code
}
```

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

BMX-07 | NEW SYNTAX FOR EMITTING EVENTS

Category	Severity	Location	Status
Language Specific	● Informational	BMX.sol: 100~101, 113, 134, 144, 154, 164	● Acknowledged

Description

Invoking events without "emit" prefix is deprecated.

Recommendation

We recommend applying the following syntax:

```
emit OwnershipTransferred(owner, newOwner);
emit Transfer(msg.sender, _to, _value);
emit Transfer(_from, _to, _value);
emit Burn(msg.sender, _value);
emit Freeze(msg.sender, _value);
emit Unfreeze(msg.sender, _value);
```

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

BMX-08 | MISSING ERROR MESSAGES

Category	Severity	Location	Status
Coding Style	● Informational	BMX.sol: 90, 99, 106, 107, 108, 109, 118, 125, 126, 127, 128, 129, 139, 140, 149, 150, 159, 160	● Acknowledged

Description

The **require** can be used to check for conditions and throw an exception if the condition is not met. It is better to provide a string message containing details about the error that will be passed back to the caller.

Recommendation

We advise adding error messages to the linked **require** statements.

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

OPTIMIZATIONS | BMX TOKEN

ID	Title	Category	Severity	Status
389-01	Unnecessary Use Of SafeMath	Gas Optimization	Optimization	● Acknowledged

389-01 | UNNECESSARY USE OF SAFEMATH

Category	Severity	Location	Status
Gas Optimization	● Optimization	SafeMath.sol: 20; Timelock.sol: 63, 87	● Acknowledged

Description

The `SafeMath` library is used unnecessarily. With Solidity compiler versions 0.8.0 or newer, arithmetic operations will automatically revert in case of integer overflow or underflow.

```
20  library SafeMath {
```

- An implementation of `SafeMath` library is found.

```
 7      using SafeMath for uint;
```

- `SafeMath` library is used for `uint256` type in `Timelock` contract.

```
63          require(eta >= getBlockTimestamp().add(delay),
"Timelock::queueTransaction: Estimated execution block must satisfy delay.");
```

- `SafeMath.add` is called in `queueTransaction` function of `Timelock` contract.

Note: Only a sample of 2 `SafeMath` library usage in this contract (out of 3) are shown above.

Recommendation

We advise removing the usage of `SafeMath` library and using the built-in arithmetic operations provided by the Solidity programming language by setting the pragma to versions 0.8.0 or above.

Alleviation

`[CertiK]` : The team acknowledged the finding and decided to remain unchanged

APPENDIX | BMX TOKEN

Finding Categories

Categories	Description
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.
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.
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 | CERTIK

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.

CertiK | **Securing** the **Web3** World

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.