Code Security Assessment

Sandbox

Jan 12th, 2022

Table of Contents

Summary

This report has been prepared for Sandbox to discover issues and vulnerabilities in the source code of the Sandbox 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	Sandbox
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/thesandboxgame/sandbox-smart-contracts/
Commit	ac6778ecc257830cf443962e31566c069189c8ab 9d9eff206127a97e51b208a3cc9acfa0484ef2c4 f799e04bc19933da571cd0a86a64b606a89dd173

Audit Summary

Delivery Date	Jan 12, 2022
Audit Methodology	Static Analysis, Manual Review
Key Components	Gaming

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Mitigated	Resolved
Critical	0	0	0	0	0	0	0
Major	1	0	0	1	0	0	0
Medium	1	0	0	0	0	0	1
Minor	4	0	0	1	0	0	3
Informational	6	0	0	1	0	0	5
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
AHC	asset/libraries/AssetHelper.sol	49376abc7978f62fd1e72953fbe75d5356be09fa6473db4be04c5edaa32befa1
ERC	asset/libraries/ERC1155ERC721Helper.sol	f399f35c347642fab505326a19f1634018122a9dcbe65aec8d8b4e6bfd2d00a9
AAR	asset/AssetAttributesRegistry.sol	a7eb66ac6e2a110b453a5e65c850cde9d0bcbbfe16935eff583c57199c9a0d91
AMC	asset/AssetMinter.sol	c4cc865c99220407c35a3d7ca6e8e703f892906cc9e63c29c00d2b07cc2c47a5
ASA	asset/AssetSignedAuctionAuth.sol	ddc69807bc94a8d0864dbe3353b0bd4e702792bd45d538cd24b1c5aeb3069214
AUC	asset/AssetUpgrader.sol	d1581fec8a14a000a416c6335726207772a82f263cbb32b41a43289770878c81
AVC	asset/AssetV2.sol	9ae7f4ac8028192ed585efb64a922ae799d657b35f41289231743f5958346ff4
ERE	asset/ERC1155ERC721.sol	5510915caacae9e4b6d1dc331ffa08369059b9226258d213572e445a7b3500c3
IGC	catalyst/interfaces/IGemsCatalystsRegistry.sol	714e429d511faad4dd4e34c86e3dd0abef4203a058d51f1022daa420733ef7b2
CCK	catalyst/Catalyst.sol	9aa83b5f007967e081fbd40598321a5ab864c80e26d0a5ea42282a4f81a94d2d
CCM	catalyst/CollectionCatalystMigrations.sol	4fcb3efa274d14e5dc730c80f37dcc3b89072ce0f4d5239716690c04ef88a5df
GCK	catalyst/Gem.sol	4b3673afad8ca2b9f0e31a2e19c6500335869407dee4fd5a51374c041cea56e0
GCR	catalyst/GemsCatalystsRegistry.sol	092725bda1458ba4606dde119668239f8f73c97ded3ba32447352e041035c906
SER	claims/signedGiveaway/SignedERC20Giveaway.sol	9cd400a3115d0807a695c5385a3e59e629dea9eef8bdce50336b8ec0334d3569
ERT	common/BaseWithStorage/ERC20/ERC20Token.sol	95b045ca14ba2025a5f2d678c3d884847b4d53b6807f55cfbec12e7a6210bb8a

ID	File	SHA256 Checksum
ERH	common/BaseWithStorage/ERC2771Handler.sol	98f3b08790b5e5f5f8bbd00bf699b29ea30bf0017c73250738ce24b8127a6226
WAB	common/BaseWithStorage/WithAdmin.sol	020b381a262857b77a2c079a308c5ba1af16a12f8a0b1e8ed4a41dd27e5c9efa
WMB	common/BaseWithStorage/WithMinter.sol	5abee99b49456dedf0a6705876f135a59a816d2ba82508306c37be458258cc43
WPB	common/BaseWithStorage/WithPermit.sol	d5da80553c31d72af39eeeef5424b54651889506cc8facdf212d0c6da8419149
WSO	common/BaseWithStorage/WithSuperOperators.sol	c7672ef2210c6788720fc2cdbb36912564500de9b332155de9fc45818edf4dab
WUB	common/BaseWithStorage/WithUpgrader.sol	6cade310be0f69cb2e331995e678e5b07ff96958173a25f6dfb37a7e48dc4316
IAA	common/interfaces/IAssetAttributesRegistry.sol	b682b2b1ba0cd860c179ea820ab87c26799704ff24b1a3e6d5ed9165f3cb1f6c
IAM	common/interfaces/IAssetMinter.sol	9b6a515fb6167a77e3ec8e084dc8922cc77f8fa9e22312281015e9993d8c6a56
IAT	common/interfaces/IAssetToken.sol	9d1b706e70c51eb93b262017a861c53046978247312ac9f9b525a08495d8575e
IAU	common/interfaces/IAssetUpgrader.sol	17bd397358d1dfedc2a5204ff52d5ee4962dad0113b67ae6eed486354353ae67
IAC	common/interfaces/IAttributes.sol	2bd3ce8011577e3271b7597d42ec0efb21e3c40df887d60607acebf04c658e33
IER	common/interfaces/IERC20Extended.sol	349a855f29e62c1139028c5b839e409a32898ad5a9fc113334cde9cd6254e6b7
PAV	polygon/child/asset/PolygonAssetV2.sol	2a9317720f1c16e4372111c43ae3c68b467dfa01fdb91b93923c522def209083

Overview

The Sandbox has created a set of contracts related to a video game. In the context of this audit, the following contracts were audited:

- Contracts `Catalyst`, `Gem` : Respectively represent the Catalyst and Gem in the context of the video game.
- Contract `AssetV2`, `PolygonAssetV2` : Respectively represent an Ethereum asset and a Polygon Asset in the context of the video game.
- Contract `AssetAttributesRegistry` : Allows setting the gems and catalysts of an asset.
- Contract `GemsCatalystsRegistry` : Contract managing the Gems and Catalysts.
- Contract `AssetMinter` : Used to mint Asset with Catalyst, Gems and Sand.
- Contract `AssetUpgrader`: Used to upgrade Asset with Catalyst, Gems and Sand.
- Contract `CollectionCatalystMigrations`: Contract performing migrations for collections.
- Contract `AssetSignedAuctionAuth`: Contract allowing users to claim offers with signatures mechanisms (EIP712 for example).
- Contract `SignedERC20Giveaway`: Contract used to distribute asset rewards to users.

External Dependencies

There are a few depending injection contracts or addresses in the current project:

- For contract `AssetAttributesRegistry`: `_gemsCatalystsRegistry`, `_admin`, `_minter` and `_upgrader`.
- For contract `AssetMinter`: `_registry`, `_asset`, `_gemsCatalystsRegistry` and `trustedForwarder`.
- For contract `AssetSignedAuctionAuth`: `_asset`, `_feeCollector` and `_admin`.
- For contract `AssetUpgrader`: `_registry`, `_sand`, `_asset`, `_gemsCatalystsRegistry`, `feeRecipient` and `trustedForwarder`.
- For contract `AssetV2`: `trustedForwarder`, `admin`, `bouncerAdmin`, `predicate` and `assetRegistry`.
- For contract `Catalyst`: `admin` and `operator`.
- For contract `CollectionCatalystMigrations`: `_oldRegistry`, `_asset`, `_registry` and `_admin`.
- For contract `GemsCatalystsRegistry`: `_admin` and `trustedForwarder`.
- For contract `SignedERC20Giveaway`: `trustedForwarder_` and `defaultAdmin_`.
- For contract `PolygonAssetV2`: `trustedForwarder`, `admin`, `bouncerAdmin`, `assetRegistry` and `_childChainManager`.

We assume these vulnerable actors and implementing proper logic to collaborate with the current project.

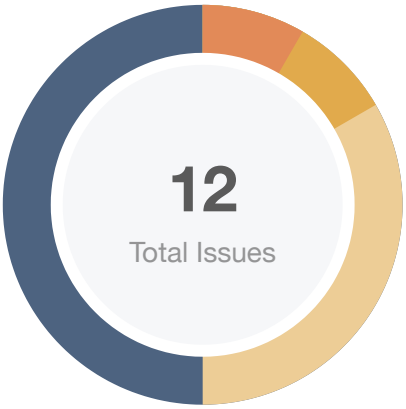
Privileged Roles

To set up the project correctly, improve overall project quality and preserve upgradability, the following roles are adopted in the codebase:

- The `owner` role is adopted in contract `AssetMinter` to modify some variables, add minters, and mint assets.
- The `owner` role is adopted in contracts `AssetMinter`, `AssetUpgrader` and `GemsCatalystsRegistry` contracts to configure the ERC2771 forwarder.
- The `Super Operator` role is adopted in contract `GemsCatalystsRegistry`, this role can burn Catalysts or Gems.
- The `_minter` and `_upgrader` roles are adopted in contract `AssetAttributesRegistry` to set Catalysts and Gems for an asset.
- The `_upgrader` role is adopted in contract `AssetAttributesRegistry` to add Gems to an existing list of Gems for an asset.
- The `_admin` and `overLayerDepositor` roles are adopted in contract `AssetAttributesRegistry`, to set the Catalysts and Gems when an asset goes over layers.
- The `_admin` role is adopted in contract `AssetAttributesRegistry`, to configure the `overLayerDepositor` address.
- The `_admin` role is adopted in contract `CollectionCatalystMigrations`, to migrate the Catalysts for assets, and to define `AssetAttributesRegistry` contract address.
- The `_admin` role is adopted in contract `GemsCatalystsRegistry`, to add Gems and Catalysts.
- The `DEFAULT_ADMIN_ROLE` role is adopted in contract `SignedERC20Giveaway`, to revoke claims, pause and unpause the contract.
- The `_bouncerAdmin` role is adopted in the contract `ERC1155ERC721` to enable/disable the `bouncer` role, which can mint tokens and update an NFT with new features.
- The `_superOperators` role is adopted in the contract `ERC1155ERC721` to approve operators to manage users' tokens and change the creatorship.

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community. Any plan to invoke the aforementioned functions should be also considered to move to the execution queue of the `Timelock` contract.

Findings



Critical	**0** (0.00%)
Major	**1** (8.33%)
Medium	**1** (8.33%)
Minor	**4** (33.33%)
Informational	**6** (50.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
GLOBAL-01	Centralization Risk	**Centralization / Privilege**	**Major**	Acknowledged
GLOBAL-02	Third Party Dependencies	Volatile Code	Informational	Acknowledged
GLOBAL-03	Missing Emit Events	Coding Style	Informational	Resolved
AAR-01	Typo in the Contract	Coding Style	Informational	Resolved
ASA-01	Usage of `transfer()` For Sending Ether	Volatile Code	Minor	Resolved
ASA-02	Redundant Statement	Coding Style	Informational	Resolved
AUC-01	Unchecked Value of ERC-20 `transferFrom()` Call	Volatile Code	Minor	Resolved
CCK-01	Risk For Weak Randomness	Logical Issue	Minor	Acknowledged
ERE-01	Missing Zero Address Validation	Coding Style	Informational	Resolved
GCR-01	Absence of function to undo the effects of `setgemsandcatalystsmaxallowance()`	Logical Issue	Medium	Resolved
GCR-02	Gas Consumption in `setgemsandcatalystsmaxallowance()`	Logical Issue	Minor	Resolved
WAB-01	Missing Zero Address Validation	Coding Style	Informational	Resolved

GLOBAL-01 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	Major	Global	Acknowledged

Description

In the contract `AssetMinter`, the role `owner` has the authority over some functions.

Especially, the `owner` can call:

- `setCustomMintingAllowance()` : Add/remove minters.
- `mintCustomNumberWithCatalyst()` : Mint assets using one catalyst.

In addition, the `owner` call following functions to update setting parameters:

- `addOrReplaceQuantitiyByCatalystId()`: Update `quantitiesByCatalystId[]` array.
- `addOrReplaceAssetTypeQuantity()`: Update `quantitiesByAssetTypeId[]` array.
- `setNumberOfGemsBurnPerAsset()`: Update the value of `numberOfGemsBurnPerAsset`.
- `setNumberOfCatalystsBurnPerAsset`: Update the value of `numberOfCatalystBurnPerAsset`.
- `setGemsFactor()`: Update the value of `gemsFactor`.
- `setCatalystsFactor()`: Update the value of `catalystsFactor`.

In the contracts `AssetMinter`, `AssetUpgrader`, and `GemsCatalystsRegistry`, the role `owner` has the authority over the following function:

- `setTrustedForwarder()` : Configure the ERC2771 forwarder. As per the EIP2771 documentation, "A bad forwarder may allow forgery of the msg.sender returned from _msgSender() and allow transactions to appear to be coming from any address.".

In the contract `AssetSignedAuctionAuth`, the `_admin` role has the authority to set fee parameters of the project.

In the contract `Catalyst`, the `onlyAdmin` role has the authority to upgrade the attributes contract.

In the contract the `CollectionCatalystMigrations`, the `_admin` role has the authority over the following functions:

- `migrate()`: Migrate the catalyst for a collection of assets
- `batchMigrate()`: Migrate the catalysts for a batch of assets.

- `setAssetAttributesRegistryMigrationContract()`: Set the registry migration contract.

In the contract `GemsCatalystsRegistry`, the "SuperOperator" has the authority to burn Catalysts or Gems from a given address.

In the contract `GemsCatalystsRegistry`, the `_upgrader` role has the authority over the following functions:

- `setCatalyst()`: Set the Catalysts and Gems for an asset.
- `addGems()`: Add Gems to an existing list of Gems for an asset.

In the contract `GemsCatalystsRegistry`, the `_minter` role has the authority to set the Catalysts and Gems for an asset.

In the contract `GemsCatalystsRegistry`, the `_admin` role has the authority over the following functions:

- `setCatalystWhenDepositOnOtherLayer()`: Set the Catalysts and Gems when an asset goes over layers.
- `setMigrationContract()`: Set the migration contract address.
- `setOverLayerDepositor()`: Set the `overLayerDepositor` role, which can invoke `setCatalystWhenDepositOnOtherLayer()` function.

In the contract `SignedERC20Giveaway`, the `DEFAULT_ADMIN_ROLE` has the authority to revoke claims, pause and unpause the contract.

In the contract `WithUpgrader`, the `_admin` role has the authority over the following functions:

- `changeUpgrader()`: Change the Upgrader address.

In the contract `WithMinter`, the `_admin` role has the authority over the following functions:

- `changeMinter()`: Change the Minter address.

In the contract `ERC20Token`, the `_admin` role has the authority over the following functions:

- `mint()`: Mint tokens.

In the contract `Catalyst`, the `_admin` role has the authority over the following functions:

- `changeAttributes()`: Modify the `Attributes` contract.

In the contract `ERC1155ERC721`, the `_superOperators` role has the authority over the following functions:

- `transferCreatorship()`: transfer the creatorship to a given address.

- `setApprovalForAllFor()`: Approve the `operator` to manage all the tokens of a given address.
- `setApprovalForAllFor()`: return `true` to allow the "superOperators" manage the tokens.

In the contract `ERC1155ERC721`, the `_bouncerAdmin` role has the authority to enable/disable a `bouncer` role, which can mint tokens and update an NFT with new features.

Any compromise to the privileged account may allow the hacker to take advantage of this and rig the game.

Recommendation

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 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:

Short-Term: Time-lock & Multi-sig

- 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;

Long-Term: DAO

- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

[**The Sandox team**]
All the admin roles are bound to The Sandbox gouvernance multisig on layer 1. Since we're migrating to Polygon, we are not comfortable enough to use time-lock yet, but the privileged roles will definitly move to a multisig. We are planning to introduce a DAO next year for all the gaming aspects.

GLOBAL-02 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	Informational	Global	Acknowledged

Description

The contract is serving as the underlying entity to interact with third-party Polygon Bridge protocols and the forwarders specified in ERC2771. The scope of the audit treats 3rd party entities as black boxes and assumes their functional correctness.

However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets.

Recommendation

We understand that the business logic of Sandbox requires interaction with Polygon Bridge and forwarders. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

[**The Sandox team**]
We are aware of the trust we provide to Polygon and trusted forwarder like Biconomy.

GLOBAL-03 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	Global	Resolved

Description

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

For example,

In the contract `AssetMinter`, function `setCustomMintingAllowance()` should emit an event to inform the community when adding/removing a minter.

In the contracts `AssetMinter`, `AssetUpgrader`, and `GemsCatalystsRegistry`, function `setTrustedForwarder()` should emit an event to inform the community when updating a forwarder.

In the contact `CollectionCatalystMigrations`, function `batchMigrate` should emit an event to inform the community when migrating.

Recommendation

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

Alleviation

The team heeded the advise and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

AAR-01 | Typo In The Contract

Category	Severity	Location	Status
Coding Style	Informational	asset/AssetAttributesRegistry.sol: 197	Resolved

Description

The linked comment statement contains a typo in its body, namely `migratcion` should be `migration`.

Recommendation

It is advised to modify the typo.

Alleviation

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

ASA-01 | Usage Of `transfer()` For Sending Ether

Category	Severity	Location	Status
Volatile Code	Minor	asset/AssetSignedAuctionAuth.sol: 351, 353, 355	Resolved

Description

After EIP-1884 was included in the Istanbul hard fork, it is not recommended to use `.transfer()` 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()` calls are 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

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

ASA-02 | Redundant Statement

Category	Severity	Location	Status
Coding Style	Informational	asset/AssetSignedAuctionAuth.sol: 107	Resolved

Description

As of Solidity v0.8.0, arithmetic operations revert on underflow and overflow by default. Therefore, the linked statements do not perform actual functionality and thus can be safely omitted.

Recommendation

It is recommended to remove the aforementioned redundant statement.

Alleviation

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

AUC-01 | Unchecked Value Of ERC-20 `transferFrom()` Call

Category	Severity	Location	Status
Volatile Code	Minor	asset/AssetUpgrader.sol: 137	Resolved

Description

In `AssetUpgrader`, L137, the linked `transferFrom()` invocation does not check the return value of the function call which should yield a `true` result in case of a proper ERC-20 implementation.

Recommendation

As many tokens do not follow the ERC-20 standard faithfully, they may not return a `bool` variable in this function's execution meaning that simply expecting it can cause incompatibility with these types of tokens. Instead, it is recommended that OpenZeppelin's `SafeERC20.sol` implementation is utilized for interacting with the `transferFrom()` functions of ERC-20 tokens. The OZ implementation optionally checks for a return value rendering compatible with all ERC-20 token implementations.

Alleviation

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

CCK-01 | Risk For Weak Randomness

Category	Severity	Location	Status
Logical Issue	Minor	catalyst/Catalyst.sol: 50~51	Acknowledged

Description

A Gem gives between 1 and 25 attributes points to an ASSET. The number of points needs to be random to ensure fairness in the game.

Within the dependancy `DefaultAttributes`, the amount of attribute points is computed as followed:

```
67  function _computeValue(
68      uint256 assetId,
69      uint256 gemId,
70      bytes32 blockHash,
71      uint256 slotIndex,
72      uint32 min
73  ) internal pure returns (uint32) {
74      return min + uint16(uint256(keccak256(abi.encodePacked(gemId, assetId,
blockHash, slotIndex))) % (26 - min));
75  }
```

A hash is computed and converted into a `uint16`. This computation is not using an external source of randomness.

The parameters to compute the hash are deterministics:

```
/// @param assetId The id of the asset.
/// @param gemId The id of the gem.
/// @param blockHash The blockHash from the gemEvent.
/// @param slotIndex Index of the current gem.
```

Because the computation of the hash uses on-chain data, if the attacker knows the `gemId`, `assetId`, `blockHash` and `slotIndex`, he can determine in advance how many attribute points his ASSET would receive with a particular Gem.

An attacker could exploit this mechanism by creating an important amount of ASSETs with the function `mintWithoutCatalyst()`. This would create a lot of different `assetId`. With the different `assetId` and the information about the Gem, he can recompute the amount of attribute points thanks to the function

`_computeValue()`. He could therefore find combinations of Gem + Assets that would give the maximum amount of attribute points.

Recommendation

It is recommended to use an external source of randomness like Chainlink VRF for generating random numbers.

Alleviation

[**The Sandox team**]
We are aware of the weakness of this randomness. This is the historical behaviour of the attributes and we cannot change it for now. The team is aware and decided to take the risk induced. Keep in mind that mintWithoutCatalyst would only produce an asset without any catalyst & gems so you cannot abuse the attributes with this method. The user could use mintWithCatalyst that will cost him catalyst gems tokens for each try.

ERE-01 | Missing Zero Address Validation

Category	Severity	Location	Status
Coding Style	Informational	asset/ERC1155ERC721.sol: 86~90	Resolved

Description

Address should be checked before assignment to make sure it is not zero addresses. In case the zero address is passed for the functions `changeBouncerAdmin()`, it would be impossible to recover the `_bouncerAdmin` accesses.

Recommendation

Consider adding a zero check. As below:

```
require(newBouncerAdmin != address(0),'new admin can't be zero address');
```

Alleviation

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

GCR-01 | Absence Of Function To Undo The Effects Of `setgemsandcatalystsmaxallowance()`

Category	Severity	Location	Status
Logical Issue	Medium	catalyst/GemsCatalystsRegistry.sol: 180~188	Resolved

Description

The `setGemsandCatalystsMaxAllowance()` is a function called by the users to give control of their gems/catalysts to the `GemCatalystsRegistry` contract. However, the equivalent reverse function is not implemented.

It is difficult for users having called this function to remove control of the contract over their gems/catalysts.

If a `Super Operator` account got compromised by an attacker, the attacker could take control over the gems/catalysts of the impacted users.

At the same time, those users would not be able to easily revoke the authorizations given by `setGemsandCatalystsMaxAllowance()`.

Recommendation

It is recommended to setup a function so users can easily revoke allowances given to the `GemCatalystsRegistry` contract.

Alleviation

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

GCR-02 | Gas Consumption In `setgemsandcatalystsmaxallowance()`

Category	Severity	Location	Status
Logical Issue	Minor	catalyst/GemsCatalystsRegistry.sol: 180~188	Resolved

Description

Calling the function `setGemsandCatalystsMaxAllowance()` will parse all existing Gems and Catalysts in the `GemsCatalystsRegistry`. If the arrays are big enough, the transaction might always fail due to reaching the limit of gas consumption.

```
180     function setGemsandCatalystsMaxAllowance() external {
181         for (uint256 i = 0; i < _gems.length; i++) {
182             _gems[i].approveFor(_msgSender(), address(this), ~uint256(0));
183         }
184 
185         for (uint256 i = 0; i < _catalysts.length; i++) {
186             _catalysts[i].approveFor(_msgSender(), address(this), ~uint256(0));
187         }
188     }
```

According to the whitepaper, the array length of Gems and Catalysts should be limited. However, the `_admin` role is able to call `addGemsAndCatalysts` to push elements into the arrays.

Recommendation

It is recommended to be careful when adding new Gems or Catalysts to avoid the "out of gas" error. For example, adding an upper bound for the total amount of Gems or Catalysts.

Alleviation

The team heeded the advice and resolve this issue in the commit f799e04bc19933da571cd0a86a64b606a89dd173.

WAB-01 | Missing Zero Address Validation

Category	Severity	Location	Status
Coding Style	Informational	common/BaseWithStorage/WithAdmin.sol: 26~30	Resolved

Description

Address should be checked before assignment to make sure it is not zero addresses. In case the zero address is passed for the functions `changeAdmin()`, it would be impossible to recover the `_admin` accesses.

Recommendation

Consider adding a zero check. As below:

```
require(newAdmin != address(0), 'new admin can't be zero address');
```

Alleviation

[**The Sandox team**]
We are aware of that. We actually want to be able to give up on the admin role.

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.

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.

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.