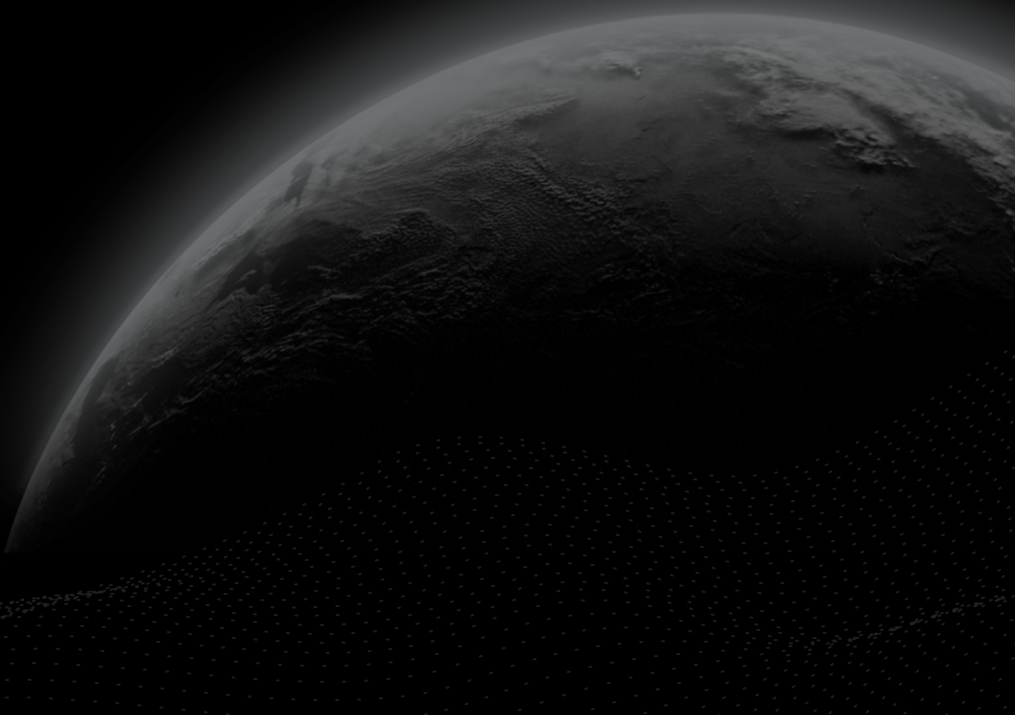




Security Assessment

Gala Games - GalaV2

CertiK Verified on Apr 20th, 2023





Certik Verified on Apr 20th, 2023

Gala Games - GalaV2

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

Executive Summary

TYPES

GameFi

ECOSYSTEM

Ethereum (ETH)

METHODS

Manual Review, Static Analysis

LANGUAGE

Solidity

TIMELINE

Delivered on 04/20/2023

KEY COMPONENTS

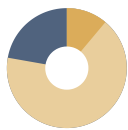
N/A

CODEBASE

Deployed Contract (ETH):

[0xd1d2eb1b1e90b638588728b4130137d262c87cae](#)[...View All](#)

Vulnerability Summary



9

Total Findings

9

Resolved

0

Mitigated

0

Partially Resolved

0

Acknowledged

0

Declined

0

Unresolved

0 Critical

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.

0 Major

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 Resolved



Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.

6 Minor

6 Resolved



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.

2 Informational

2 Resolved



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 | GALA GAMES - GALAV2

I **Summary**

[Executive Summary](#)

[Vulnerability Summary](#)

[Codebase](#)

[Audit Scope](#)

[Approach & Methods](#)

I **Decentralization Efforts**

[Description](#)

[Upgradeability](#)

[GalaV2](#)

[Recommendation](#)

[Short Term](#)

[Long Term](#)

[Permanent](#)

[Alleviation](#)

[Update 04/20/2022](#)

I **Findings**

[GVG-03 : Zero Address Can be `blocklisted`](#)

[GVG-04 : Inheritance `PausableUpgradeable` is not Initialized](#)

[GVG-05 : Inconsistent Use of `msg.sender` and `\\_msgSender\(\)`](#)

[GVG-06 : Discussion on `MINTER\\_ROLE`](#)

[GVG-07 : Minters can be added and removed when contract is `\\_paused`](#)

[GVG-08 : Inconsistency Between `mint\(\)` and `mintBulk\(\)`](#)

[GVG-09 : Missing Zero Address Validation](#)

[GVG-10 : `totalMinted` vs. `totalSupply`](#)

[GVG-11 : The keyword `super` is unnecessary](#)

I **Optimizations**

[GVG-12 : Costly Operation Inside Loop](#)

[GVG-13 : Redundant `notBlocklisted` and `whenNotPaused` Checks](#)

[GVG-14 : Variable Can Be Declared As `constant`](#)

[GVV-01 : Redundant Checks on Admin Roles](#)

I **Appendix**

Disclaimer

CODEBASE | GALA GAMES - GALAV2

Repository

Deployed Contract (ETH): [0xd1d2eb1b1e90b638588728b4130137d262c87cae](#)

AUDIT SCOPE | GALA GAMES - GALAV2

1 file audited ● 1 file with Resolved findings

ID	File	SHA256 Checksum
● GVG	 projects/GALA-v2/contracts/GalaV2.sol	879f6dc230fe052398aa567daa3ba97494f1c3 f61580ca2e819bdd1b300f1d53

APPROACH & METHODS | GALA GAMES - GALAV2

This report has been prepared for Gala Games to discover issues and vulnerabilities in the source code of the Gala Games - GalaV2 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.

DECENTRALIZATION EFFORTS | GALA GAMES - GALAV2

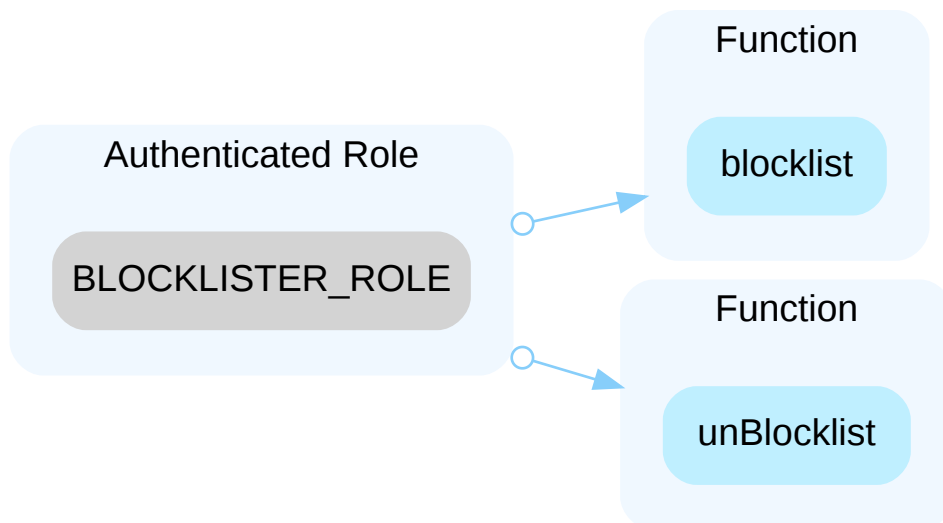
Description

Upgradeability

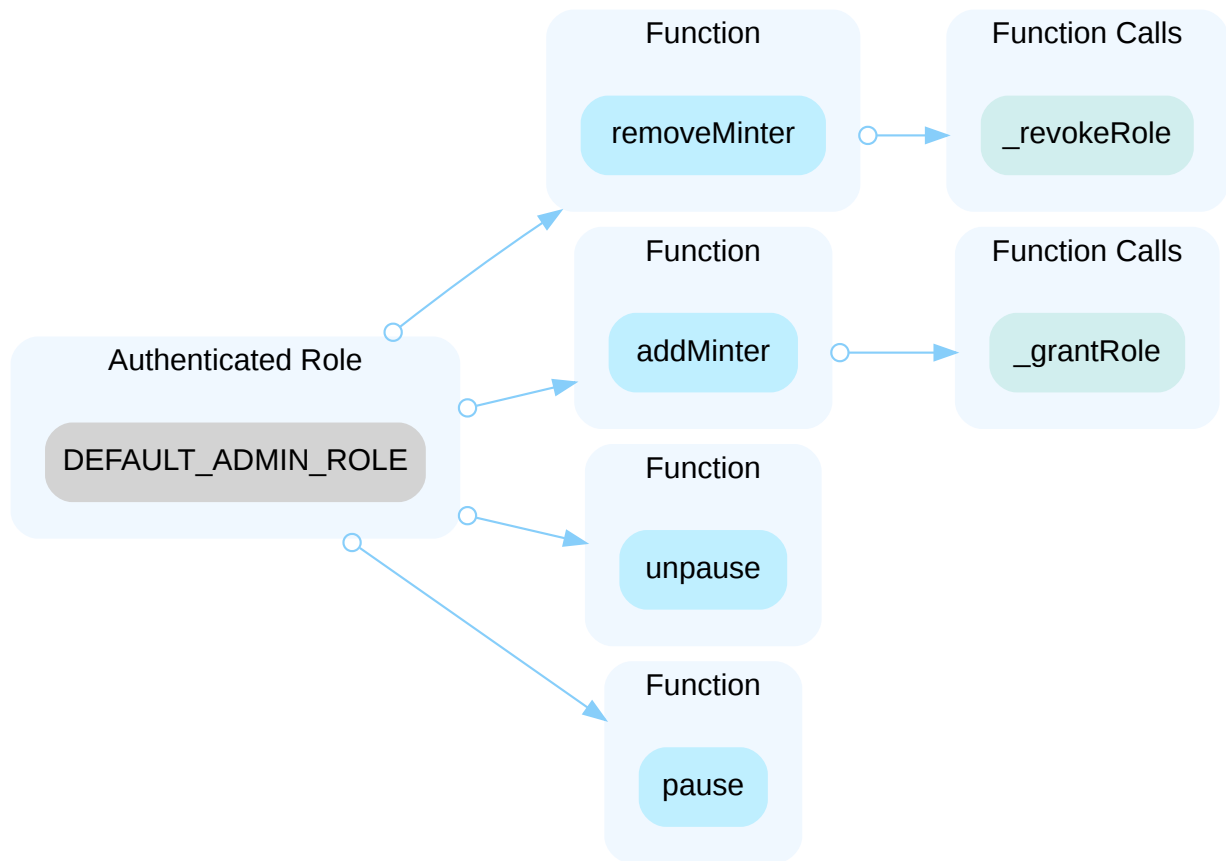
`Ga1aV2` is an upgradeable contract; the owner can upgrade the contract without the community's commitment. If an attacker compromises the account, they can change the implementation of the contract and drain tokens from the contract.

GalaV2

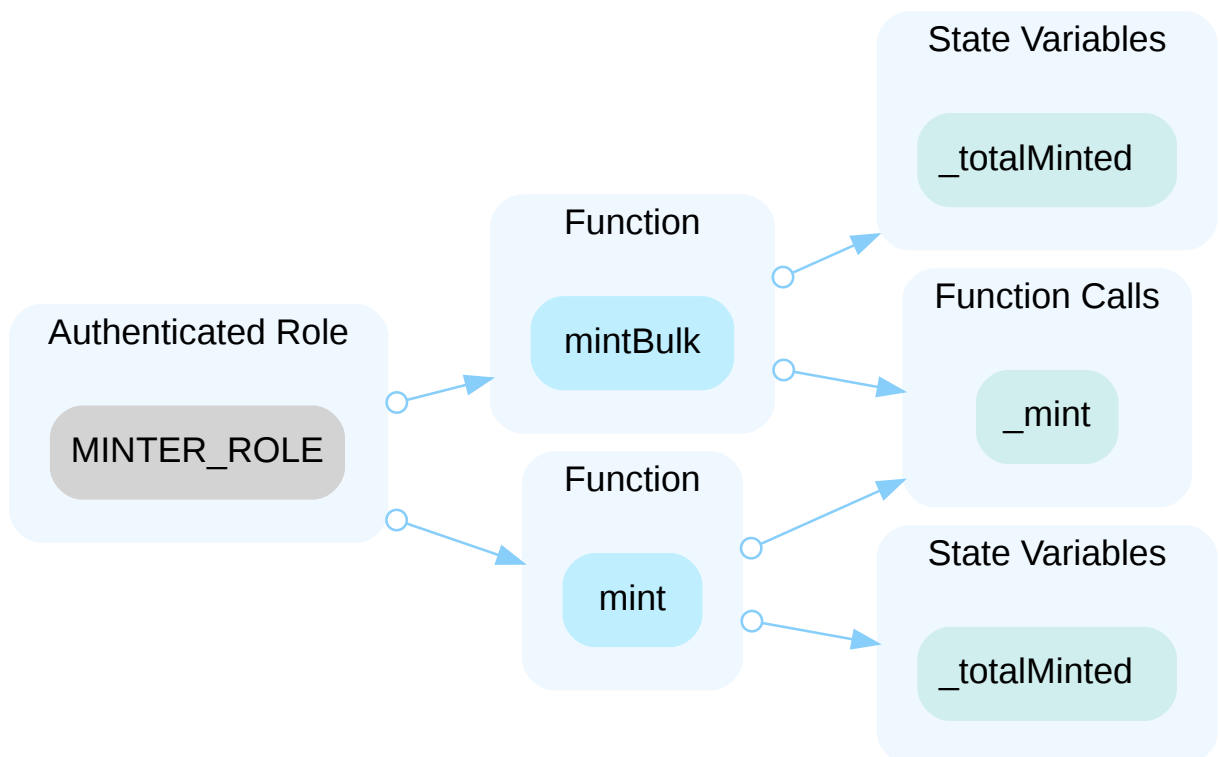
In the contract `Ga1aV2` the role `BLOCKLISTER_ROLE` has authority over the functions shown in the diagram below. Any compromise to the `BLOCKLISTER_ROLE` account may allow the hacker to take advantage of this authority and blocklist important addresses, or unblocklist malicious addresses.



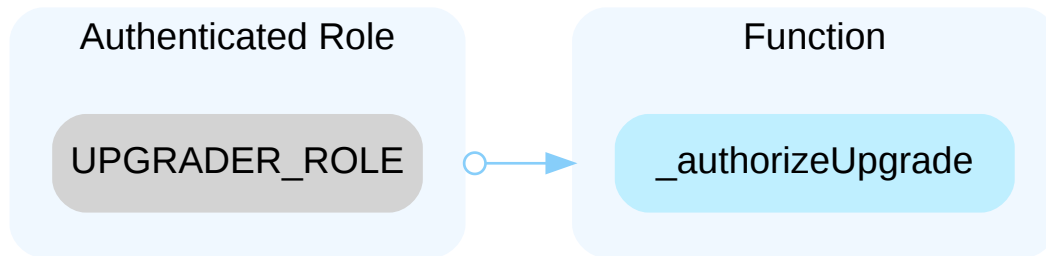
In the contract `Ga1aV2` the role `DEFAULT_ADMIN_ROLE` has authority over the functions shown in the diagram below. Any compromise to the `DEFAULT_ADMIN_ROLE` account may allow the hacker to take advantage of this authority and give themselves authority, directly or indirectly, mint tokens to a malicious address, upgrade to malicious contract logic, and add or remove accounts from the blocklist. The account may also pause the contract to keep others from using it.



In the contract `GaLaV2` the role `MINTER_ROLE` has authority over the functions shown in the diagram below. Any compromise to the `MINTER_ROLE` account may allow the hacker to take advantage of this authority and mint tokens to themselves or other accounts.



In the contract `Ga1aV2` the role `UPGRADER_ROLE` has authority over the functions shown in the diagram below. Any compromise to the `UPGRADER_ROLE` account may allow the hacker to take advantage of this authority and upgrade to malicious contract logic.



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 ($\frac{2}{3}$, $\frac{3}{5}$) 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.

I Alleviation

[certik] : The team states they will be utilizing a gnosis safe smart contract account.

They have have made the following adjustments in the codebase to prepare for this transition.

- Separated accounts for different roles in the initialize function. (minter, upgrader and blocklister);
- Added the ability to transfer roles to different account on each role type

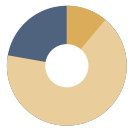
Update 04/20/2022

The team deployed `GalaV2` at address [0xd1d2eb1b1e90b638588728b4130137d262c87cae](#) on Ethereum Mainnet.

The roles are set as follows:

- `BLOCKLISTER_ROLE` : [0xcf3b0d37f639e81c0a35ee083dac09504ab3ee3a](#)
- `DEFAULT_ADMIN_ROLE` : [0xdae4249fa57610723b1f1b1c0379897aed4c9f26](#)
- `MINTER_ROLE` : [0xec78b484a28a73f4f1cff23b9cd31aa12c1d9b4e](#)
- `UPDGRADER_ROLE` : [0xcf3b0d37f639e81c0a35ee083dac09504ab3ee3a](#)

FINDINGS | GALA GAMES - GALAV2



9

Total Findings

0

Critical

0

Major

1

Medium

6

Minor

2

Informational

This report has been prepared to discover issues and vulnerabilities for Gala Games - GalaV2. Through this audit, we have uncovered 9 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
GVG-03	Zero Address Can Be <code>blocklisted</code>	Logical Issue	Medium	Resolved
GVG-04	Inheritance <code>PausableUpgradeable</code> Is Not Initialized	Volatile Code	Minor	Resolved
GVG-05	Inconsistent Use Of <code>msg.sender</code> And <code>_msgSender()</code>	Inconsistency	Minor	Resolved
GVG-06	Discussion On <code>MINTER_ROLE</code>	Control Flow	Minor	Resolved
GVG-07	Minters Can Be Added And Removed When Contract Is <code>_paused</code>	Control Flow	Minor	Resolved
GVG-08	Inconsistency Between <code>mint()</code> And <code>mintBulk()</code>	Inconsistency	Minor	Resolved
GVG-09	Missing Zero Address Validation	Volatile Code	Minor	Resolved
GVG-10	<code>_totalMinted</code> Vs. <code>_totalSupply</code>	Logical Issue, Coding Style	Informational	Resolved
GVG-11	The Keyword <code>super</code> Is Unnecessary	Coding Style	Informational	Resolved

GVG-03 | ZERO ADDRESS CAN BE blocklisted

Category	Severity	Location	Status
Logical Issue	● Medium	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 148~149, 168~169	● Resolved

Description

The `BLOCKLISTER_ROLE` can `blocklist()` the zero address. This would prevent all mints and burns and it may not be immediately clear what is causing the issue.

Recommendation

We recommend adding a check that none of the addresses in the input parameter `_accounts` is `address(0)` for function `blocklist()`.

Alleviation

[CertiK]: The team heeded the recommendation and made changes resolving the finding.

GVG-04 | INHERITANCE `PausableUpgradeable` IS NOT INITIALIZED

Category	Severity	Location	Status
Volatile Code	● Minor	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 21~22, 45~46	● Resolved

Description

Inherited contract `PausableUpgradeable` is not initialized through call to `__Pausable_init()` in the `initialize()` function.

Recommendation

We recommend calling this function so that `_paused` is initialized explicitly as `false`.

Alleviation

[CertiK]: The team heeded the recommendation and made changes resolving the finding.

GVG-05 | INCONSISTENT USE OF `msg.sender` AND `_msgSender()`

Category	Severity	Location	Status
Inconsistency	● Minor	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 52~55, 63~64, 74~75, 94~95, 123~124, 126~127, 133~134, 150~151, 192~193, 204~205	● Resolved

Description

The child contract uses both `msg.sender` and `_msgSender()` in its functions, while the inherited contracts consistently use `_msgSender()`.

If `_msgSender()` is ever updated to be anything other than `msg.sender`, this may cause inconsistencies.

Recommendation

We recommend the consistent use of `_msgSender()` throughout the child contract.

Alleviation

[CertiK]: The team heeded the recommendation and made changes resolving the finding.

GVG-06 | DISCUSSION ON MINTER_ROLE

Category	Severity	Location	Status
Control Flow	Minor	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 62~63, 73~74, 168~169, 229~230, 237~238	Resolved

Description

There is custom functionality created for accomplishing granting or revoking the `MINTER_ROLE`: `addMinter()` and `removeMinter()`.

Moreover, the functions `mint()` and `mintBulk()` also perform specific checks to ensure that the `MINTER_ROLE` is also not blocklisted.

Please provide more information on the types of accounts you plan to grant the `MINTER_ROLE`. Is it within the design plans of the project to allow for multiple accounts to take this role; will general users be given this privilege, where there may be a need to blocklist bad actors?

Recommendation

We recommend providing more information on the design plans around the `MINTER_ROLE`.

Additionally, we recommend updating function logic so that no single account can be both blocklisted and a privileged role such as `MINTER_ROLE`, `BLOCKLISTER_ROLE`, or `UPGRADER_ROLE` simultaneously.

Alleviation

[Gala Games]: "the plan for the minter role is to be an EOA since our bridge is the minter. I have looked into using the smart account with gnosis safe but there is no way to do this through their api in exception of using their website.

I have also added the check to the blocklist logic so it cannot blocklist address with roles."

[CertiK]: From the Gala Games team response, it is noted that the `MINTER_ROLE` is a privileged status that is meant to be used with their bridge.

Note that all roles can still be granted to accounts that have been blocklisted through the function `grantRole()`.

We recommend translating the modifier checks `notBlocklisted` and `notZeroAddress` to the `grantRole()` function and removing the modifier from specialty functions `addMinter()`, `addUpgrader()`, `addBlocklister()`, and `transferAdmin()`.

We also make a note here that newly added `transferAdmin()` function can only be used successfully if every time an admin is to be added, another is also to be revoked. Otherwise, `grantRole()` will need to be used directly.

Update 02/15/2023

[certik] : The team made changes resolving this finding.

GVG-07 | MINTERS CAN BE ADDED AND REMOVED WHEN CONTRACT IS `_paused`

Category	Severity	Location	Status
Control Flow	● Minor	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 229~230, 237~238	● Resolved

Description

Functions `addMinter()` and `removeMinter()`, which grant the `MINTER_ROLE`, can only be called when the contract is not `_paused`; however, `grantRole()` and `revokeRole()` may still be called for any address and role, even when the contract is `_paused`.

Recommendation

We recommend modifying the contract logic so that the corresponding functions perform consistently when the contract is paused or unpaused, in line with the intended design.

Alleviation

[Certik]: The team made updates to functions `addMinter()`, `removeMinter()`, `grantRole()`, and `revokeRole()`.

They also added functions

- `addUpgrader()` / `removeUpgrader()`;
- `addBlocklister()` / `removeBlocklister()`;
- `transferAdmin()`

All functions can now only be called when the contract is not paused.

Note that a privileged role may still call `renounceRole()` to renounce their own role even when the contract is paused. If this is intended for the design of the contract, no changes need to be made.

GVG-08 | INCONSISTENCY BETWEEN `mint()` AND `mintBulk()`

Category	Severity	Location	Status
Inconsistency	● Minor	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 60~61, 84~85	● Resolved

Description

Function `mint()` allows for a zero input `amount`, while each `amounts[i]` must be positive in `mintBulk()`

Recommendation

We recommend updating the corresponding functions to be consistent, in line with the design of the project.

Alleviation

[certik]: The team made changes resolving the finding.

Now, `mint()` requires that `amount` is nonzero.

GVG-09 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	Minor	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 229~230	Resolved

Description

The cited address input is missing a check that it is not `address(0)`.

Recommendation

We recommend adding a check that the passed-in address is not `address(0)` to prevent unexpected errors.

Alleviation

[certik]: The team made changes resolving this finding.

GVG-10 | `_totalMinted` VS. `_totalSupply`

Category	Severity	Location	Status
Logical Issue, Coding Style	● Informational	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 68~69, 86~87, 127~128, 138~139, 157~160	● Resolved

Description

Gala Token - Version 1

In version 1 of the `Gala` token, the `_totalMinted` value is only updated during calls to `mint()`, where it is updated as:

```
_totalMinted = _totalMinted.add(amount);
```

In the hook `_beforeTokenTransfer()`, the update to this value is checked against the set `_cap` when minting:

```
require(totalMinted().add(amount) <= cap(), "ERC20Capped: cap exceeded");
```

Since `_totalMinted` is only updated in `mint()` and is checked against the `_cap` value, this means that only 50 billion `GALA` will ever be minted, and that as users burn the token, the `_totalSupply` will monotonically decrease towards 0.

GalaV2

While `_totalMinted` is still updated and checked during functions `mint()` and `mintBulk()` in the same way for `GalaV2`, the value `_totalMinted` is now updated in functions `burn()` and `burnFrom()` as follows:

```
_totalMinted = _totalMinted - amount;
```

This means that as tokens are burned, an equivalent amount becomes available again to be minted.

As such, this new update to `_totalMinted` makes the value consistent with `_totalSupply` throughout the lifetime of the `GalaV2` token. Indeed, the `_cap` on `_totalMinted` is also a `_cap` on the `_totalSupply`, making the values equivalent in all respects, within the current contract's set up.

Recommendation

We recommend clarifying whether there are future plans to distinguish between `_totalMinted` and `_totalSupply` or whether there is a necessity for having two different variables recording the same value. Otherwise, consider removing the `_totalMinted` and checking `_totalSupply` against the `_cap` during minting to simplify the token model.

■ Alleviation

[Certik]: The team made changes resolving this finding.

The token no longer includes a `_totalMinted` variable, and instead relies on comparing the `_totalSupply` value against the `_cap` during minting.

To summarize, since `_totalSupply` is updated for both minting and burning, when tokens are burned, this allows more tokens to be minted, as long as the amount is within the `_cap` upper bound.

GVG-11 | THE KEYWORD `super` IS UNNECESSARY

Category	Severity	Location	Status
Coding Style	● Informational	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 245~247, 249~251, 253~255	● Resolved

Description

The use of the `super` keyword is not necessary since there is no override and there is a well-defined path to the internal `_pause()` and `_unpause()` functions inherited from `ERC20PausableUpgradeable`, and to the public `hasRole()` function from `AccessControlUpgradeable`.

Recommendation

We recommend removing the keyword `super` when calling the cited functions in base contracts `ERC20PausableUpgradeable` and `AccessControlUpgradeable`.

Alleviation

[CertiK]: The team heeded the recommendation and made changes resolving the finding.

OPTIMIZATIONS | GALA GAMES - GALAV2

ID	Title	Category	Severity	Status
GVG-12	Costly Operation Inside Loop	Gas Optimization	Optimization	● Resolved
GVG-13	Redundant <code>notBlocklisted</code> And <code>whenNotPaused</code> Checks	Gas Optimization	Optimization	● Resolved
GVG-14	Variable Can Be Declared As <code>constant</code>	Gas Optimization	Optimization	● Resolved
GVV-01	Redundant Checks On Admin Roles	Inconsistency, Gas Optimization	Optimization	● Resolved

GVG-12 | COSTLY OPERATION INSIDE LOOP

Category	Severity	Location	Status
Gas Optimization	● Optimization	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 86	● Resolved

Description

Accessing storage variables in a loop can be costly in terms of gas consumption.

```
86         _totalMinted = _totalMinted + amounts[i];
```

Recommendation

We recommend using a local variable to hold the intermediate result.

Alleviation

[CertiK]: The team made changes resolving this finding.

Now, `_totalMinted` has been removed as a state variable, so it is no longer referenced inside the loop.

GVG-13 | REDUNDANT `notBlocklisted` AND `whenNotPaused` CHECKS

Category	Severity	Location	Status
Gas Optimization	● Optimization	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 63~65, 74~75, 123~124, 133~135, 148~151	● Resolved

Description

The checks using the `notBlocklisted` and `whenNotPaused` modifiers are redundant in the following functions, since the function calls interact with hook `_beforeTokenTransfer()`, covering all possible cases:

- `mint()`;
- `mintBulk()`;
- `burn()`;
- `burnFrom()`

Recommendation

We recommend removing the redundant checks in the functions above in order to optimize code.

Alleviation

The team made changes resolving this finding.

GVG-14 | VARIABLE CAN BE DECLARED AS `constant`

Category	Severity	Location	Status
Gas Optimization	● Optimization	projects/GALA-v2/contracts/GalaV2.sol (CertiKbase): 57~58	● Resolved

Description

The variable `_cap` assigned in the initializer is never modified throughout the contract and is a hardcoded value. Instead, the variable can be initialized before the function call, with keyword `constant`, to optimize gas usage.

Recommendation

We recommend initializing state variable `_cap` as `constant`.

Alleviation

[CertiK]: The team heeded the recommendation and made changes resolving the finding.

GVV-01 | REDUNDANT CHECKS ON ADMIN ROLES

Category	Severity	Location	Status
Inconsistency, Gas Optimization	● Optimization	projects/GalaV2/contracts/GalaV2.sol (CertiKBase Update1): 240~241, 248~249, 256~257, 264~265, 272~273, 280~281, 288~289, 299~300, 310~311	● Resolved

Description

Functions `addMinter()`, `addUpgrader()`, `addBlocklist()`, and `transferAdmin()` all have redundant checks to modifier `onlyRole`. The aforementioned functions specify that only the `DEFAULT_ADMIN_ROLE` may call these functions; by default, this is the `adminRole` assigned to all roles. The redundancy also exists within functions `removeMinter()`, `removeUpgrader()`, and `removeBlocklist()`.

In the case where `DEFAULT_ADMIN_ROLE` account calls a function like `addMinter()`, the check on the `_msgSender()` confirming they are the `DEFAULT_ADMIN_ROLE` is performed twice; once at the beginning of `addMinter()` and once at the execution of `grantRole()`.

It is noted that if the contract is ever upgraded, each `role` may be assigned an `adminRole` distinct from `DEFAULT_ADMIN_ROLE`. This can be done through adding additional functionality which interacts with internal function `_setRoleAdmin()`.

For example, it is possible that another "MinterAdmin" is assigned separate from the `DEFAULT_ADMIN_ROLE` as the `adminRole` for `MINTER_ROLE`. In such a case, this "MinterAdmin" would be able to call `grantRole()` and `revokeRole()` directly, but would not be able to call function `addMinter()`, since this function specifies that only `DEFAULT_ADMIN_ROLE` may perform the call. Furthermore, `DEFAULT_ADMIN_ROLE` would cease to be able to call `addMinter()` in this instance.

Recommendation

We recommend keeping only the `onlyRole(getRoleAdmin(role))` modifier check in the `grantRole()` and `revokeRole()` functions, rather than performing multiple checks with the `hasRole` modifier in the added functions. This will optimize the codebase and prevent inconsistencies, should the contract ever be upgraded.

Alleviation

[CertiK]: The team made changes resolving the finding.

In the changes, function `transferAdmin()` was updated so that `revokeRole()` is explicitly called on the `_msgSender()`, rather than being called on a specified input parameter `_oldOwner`.

As long as the `adminRole` for the `DEFAULT_ADMIN_ROLE` is never updated, this should work as intended. If the `adminRole` for `DEFAULT_ADMIN_ROLE` is updated to a nonzero value (i.e., not the `DEFAULT_ADMIN_ROLE`), and if the `adminRole`

account does not also have `DEFAULT_ADMIN_ROLE` status, then the call will successfully complete without revoking `DEFAULT_ADMIN_ROLE` status from any account.

In this case, however, the calls in `transferAdmin()` can still be executed directly by calling `grantRole()` and `revokeRole()` directly as the `adminRole` of `DEFAULT_ADMIN_ROLE`. Additionally, the `_oldowner` address can call function `renounceRole()` themselves.

APPENDIX | GALA GAMES - GALAV2

Finding Categories

Categories	Description
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.
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 | 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.



