PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For Dragon Crypto Gaming

22 July 2022

paladinsec.co

info@paladinsec.co

Table of Contents

Disclaimer

Paladin Blockchain Security ("Paladin") has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies. Further, this audit report shall not be disclosed nor transmitted to any persons or parties on any objective, goal or justification without due written assent, acquiescence or approval by Paladin.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report. Information is provided 'as is', and Paladin is under no covenant to the completeness, accuracy or solidity of the contracts audited. In no event will Paladin or its partners, employees, agents or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

Cryptocurrencies and any technologies by extension directly or indirectly related to cryptocurrencies are highly volatile and speculative by nature. All reasonable due diligence and safeguards may yet be insufficient, and users should exercise considerable caution when participating in any shape or form in this nascent industry.

The audit report has made all reasonable attempts to provide clear and articulate recommendations to the Project team with respect to the rectification, amendment and/or revision of any highlighted issues, vulnerabilities or exploits within the contracts provided. It is the sole responsibility of the Project team to sufficiently test and perform checks, ensuring that the contracts are functioning as intended, specifically that the functions therein contained within said contracts have the desired intended effects, functionalities and outcomes of the Project team.

Paladin retains full rights over all intellectual property (including expertise and new attack or exploit vectors) discovered during the audit process. Paladin is therefore allowed and expected to re-use this knowledge in subsequent audits and to inform existing projects that may have similar vulnerabilities. Paladin may, at its discretion, claim bug bounties from third-parties while doing so.

1 Overview

This report has been prepared for Dragon Crypto Gaming's The Legend of Aurum Draconis's token contract on the Avalanche network. Paladin provides a user-centred examination of the smart contracts to look for vulnerabilities, logic errors or other issues from both an internal and external perspective.

1.1 Summary

Project Name	The Legend of Aurum Draconis
URL	https://dragoncrypto.io/
Network	Avalanche
Language	Solidity

1.2 Contracts Assessed

Name	Contract	Live Code Match
`DragonCryptoArgenti`	`0x250bdca7D1845cd543BB55E7D82dcA24D48E9f0F`	MATCH

1.3 Findings Summary

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High	0	-	-	-
Medium	0	-	-	-
Low	2	2	-	-
Informational	5	5	-	-
Total	**7**	**7**	**-**	**-**

Classification of Issues

Severity	Description
High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency.
Medium	Bugs or issues that may be subject to exploit, though their impact is somewhat limited. Issues under this classification are recommended to be fixed as soon as possible.
Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
Informational	Consistency, syntax or style best practices. Generally pose a negligible level of risk, if any.

1.3.1 DragonCryptoArgenti

ID	Severity	Summary	Status
01	LOW	Usage of a mapping type makes it difficult for users to figure out who the exact list of privileged minters are	RESOLVED
02	LOW	Governance privilege: Any governance-privileged wallet can instantly mint up to the maximum supply	RESOLVED
03	INFO	`addMinter`, `removeMinter` and `mint` can be made external	RESOLVED
04	INFO	Lack of events for `addMinter` and `removeMinter`	RESOLVED
05	INFO	Lack of validation	RESOLVED
06	INFO	Unused import: `Ownable.sol`	RESOLVED
07	INFO	Typographical errors	RESOLVED

2 Findings

2.1 DragonCryptoArgenti

`DragonCryptoArgenti` is one of the main ERC20 tokens within the Legend of Aurum Draconis game.

According to the Aurum Draconis team, this token will be used for crafting various items and be relevant for consumables and resources in general.

The token has an initial pre-mint of 15,558,706.11 tokens which are minted to their multi-signature set-up. It allows any governance mint-privileged wallet to mint additional tokens but there is a hardcoded maximum supply of 40,000,000 tokens cannot be exceeded in any way.

The breakdown of the pre-mint allocation is as follows:

- Seed and private mint: 10,160,233.1
- Liquidity mint: 2,800,000
- Ecosystem mint: 174,473.01
- Public mint: 2,424,000

These allocations are not enforced within the contract itself but have been communicated to us.

The DragonCryptoArgenti token represents only a small portion of the Aurum Draconis ecosystem. It should be noted that this audit solely covers this token. At the time of this initial token audit, Paladin has already commenced a secondary audit for the rest of the game's contracts.

2.1.1 Token Overview

Address	0x250bdca7D1845cd543BB55E7D82dcA24D48E9f0F
Token Supply	40,000,000
Decimal Places	18
Transfer Max Size	None
Transfer Min Size	None
Transfer Fees	None
Pre-mints	15,558,706.11

2.1.2 Privileged Functions

- `addMinter`
- `removeMinter`
- `setHourlyLimit`
- `setMultisig`
- `mint`
- `transferOwnership`
- `renounceOwnership`

2.1.3 Issues & Recommendations

Issue #01	**Usage of a mapping type makes it difficult for users to figure out who the exact list of privileged minters are**
Severity	LOW SEVERITY
Description	Presently, any address can be added to a set of minters. However, as this set is simply represented in a mapping, it becomes very difficult for users to get a good view of all minters that have minting privileges. The fact that no events are emitted right now makes this even more difficult to enumerate. Users might want to inspect the complete list of privileged minters for safety reasons. Making it easy for these users to do this can be valuable.
Recommendation	Consider moving the minter set to an `EnumerableSet`. This allows you to add a `getMinterLength()` and `getMinterAt` function. A user-friendly `getMinters()` function could be added as well but this of course could eventually run out of gas as it returns a dynamic number of elements.
Resolution	RESOLVED The client has moved to an `EnumerableSet`, significantly improving the contract.

Issue #02	**Governance privilege: Any governance-privileged wallet can instantly mint up to the maximum supply**
Severity	LOW SEVERITY
Description	Presently, the contract allows the multi-signature set up to add a list of privileged wallets that can call the `mint` function. Any of these wallets can mint tokens up to the maximum supply. This means that if any one of these wallets is ever compromised, it may mint a significant number of tokens, potentially compromising this aspect of the whole project.
Recommendation	Consider being extremely careful with the wallets that are are privileged to call the `mint` function. An ideal safeguard is to add hourly mint limits per privileged wallet. If more DCAR is minted than this hourly limit, mints revert. A safeguard like this reduces the potential impact of a mint wallet being compromised to a few multiples of the hourly limit and could have avoided many catastrophes for projects that had their keys stolen.
Resolution	RESOLVED The client has added both global and per-minter hourly minting limits that can solely be set by the governance multi-signature configuration. We commend the client for taking these worst-case scenarios so seriously and for safeguarding the contract against them.

Issue #03	`addMinter`, `removeMinter` and `mint` can be made external
Severity	INFORMATIONAL
Description	Functions that are not used within the contract but only externally can be marked as such with the external keyword. Apart from being a best practice when the function is not used within the contract, this can lead to a lower gas usage in certain cases.
Recommendation	Consider marking the functions mentioned above as external.
Resolution	RESOLVED

Issue #04	Lack of events for `addMinter` and `removeMinter`
Severity	INFORMATIONAL
Description	Functions that affect the status of sensitive variables should emit events as notifications.
Recommendation	Add events for the above functions.
Resolution	RESOLVED

Issue #05	Lack of validation
Severity	INFORMATIONAL

Description	The contract contains functions with parameters which are not properly validated. Having unvalidated parameters could allow the governance or users to provide variable values which are unexpected and incorrect. This could cause side-effects or worse exploits in other parts of the codebase. Consider validating the following function parameters:
	
	Line 23
	`function addMinter(address toAdd) public`
	
	Consider enforcing that the `toAdd` address has not been added yet.
	
	Line 28
	`function removeMinter(address toRemove) public {`
	
	Consider enforcing that the `toRemove` address has been added.

Recommendation	Consider validating the function parameters mentioned above.
Resolution	RESOLVED

Issue #06	Unused import: Ownable.sol
Severity	INFORMATIONAL
Location	Line 6 `import "@openzeppelin/contracts/access/Ownable.sol";`
Description	Files that are imported in a contract but not used within said contract could confuse third-party auditors. They also increase the contract length unnecessarily.
Recommendation	Consider removing the `import` to keep the contract short and simple.
Resolution	RESOLVED

Issue #07	Typographical errors
Severity	INFORMATIONAL

Description	The contract contains a number of typographic mistakes which we've enumerated below in a single issue in an effort to keep the report size reasonable.
	
	Line 9
	`//TODO - talk to dexalot and get the info about vesting`
	
	Having TODO's in production code is not desired as production code cannot be modified. Consider removing this line.
	
	Line 21
	`mapping(address => bool) public MintingPermissions;`
	
	It is considered standard Solidity practice to start variables with a lowercase.

Recommendation	Consider fixing the typographical errors.
Resolution	RESOLVED


PALADIN
BLOCKCHAIN SECURITY