FANTIUM SECURITY AUDIT REPORT

January 10, 2023

MixBytes()

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only. The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of Client. If you are not the intended recipient(s) of this document, please note that any disclosure, copying or dissemination of its content is strictly forbidden.

1.2 Security Assessment Methodology

A group of auditors are involved in the work on the audit. The security engineers check the provided source code independently of each other in accordance with the methodology described below:

1. Project architecture review:

- Project documentation review.
- General code review.
- Reverse research and study of the project architecture on the source code alone.

Stage goals
- Build an independent view of the project's architecture.
- Identifying logical flaws.

2. Checking the code in accordance with the vulnerabilities checklist:

- Manual code check for vulnerabilities listed on the Contractor's internal checklist. The Contractor's checklist is constantly updated based on the analysis of hacks, research, and audit of the clients' codes.
- Code check with the use of static analyzers (i.e Slither, Mythril, etc).

Stage goal

Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flash loan attacks etc.).

3. Checking the code for compliance with the desired security model:

- Detailed study of the project documentation.
- Examination of contracts tests.
- Examination of comments in code.
- Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit.
- Exploits PoC development with the use of such programs as Brownie and Hardhat.

Stage goal

Detect inconsistencies with the desired model.

4. Consolidation of the auditors' interim reports into one:

- Cross check: each auditor reviews the reports of the others.
- Discussion of the issues found by the auditors.
- Issuance of an interim audit report.

Stage goals

- Double-check all the found issues to make sure they are relevant and the determined threat level is correct.
- Provide the Client with an interim report.

5. Bug fixing & re-audit:

- The Client either fixes the issues or provides comments on the issues found by the auditors. Feedback from the Customer must be received on every issue/bug so that the Contractor can assign them a status (either "fixed" or "acknowledged").
- Upon completion of the bug fixing, the auditors double-check each fix and assign it a specific status, providing a proof link to the fix.
- A re-audited report is issued.

Stage goals

- Verify the fixed code version with all the recommendations and its statuses.
- Provide the Client with a re-audited report.

6. Final code verification and issuance of a public audit report:

- The Customer deploys the re-audited source code on the mainnet.
- The Contractor verifies the deployed code with the re-audited version and checks them for compliance.
- If the versions of the code match, the Contractor issues a public audit report.

Stage goals

- Conduct the final check of the code deployed on the mainnet.
- Provide the Customer with a public audit report.

Finding Severity breakdown

All vulnerabilities discovered during the audit are classified based on their potential severity and have the following classification:

Severity	Description
Critical	Bugs leading to assets theft, fund access locking, or any other loss of funds.
High	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.
Medium	Bugs that can break the intended contract logic or expose it to DoS attacks, but do not cause direct loss funds.
Low	Bugs that do not have a significant immediate impact and could be easily fixed.

Based on the feedback received from the Customer regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

Status	Description
Fixed	Recommended fixes have been made to the project code and no longer affect its security.
Acknowledged	The Customer is aware of the finding. Recommendations for the finding are planned to be resolved in the future.

1.3 Project Overview

Fantium is the platform where users can invest in athletes' NFTs. The project contains a NFT Contract and a special controller for validating users and minting tokens.

1.4 Project Dashboard

Project Summary

Title	Description
Client	Fantium
Project name	Fantium Platform
Timeline	November 22 2022 - November 29 2022
Number of Auditors	4

Project Log

Date	Commit Hash	Note
18.11.2022	cb2d97bc30c40321991fe5ab8fc798babba1610f	Commit for the audit
05.12.2022	fdb089fa02c36560645f5ae7a3ab06d63f37ee1f	Commit for the re-audit

Date	Commit Hash	Note
15.12.2022	e79ed7dddabc482c56f7828bd9a8725fbbeca2f5	Commit for the re-audit (v2)
26.12.2022	4307c73d332aeed51bca8ae0c776a992c7d9eb93	Final commit

Project Scope

The audit covered the following files:

File name	Link
FantiumNFTV1.sol	FantiumNFTV1.sol
FantiumMinterV1.sol	FantiumMinterV1.sol
DefaultOperatorFiltererUpgradeable.sol	DefaultOperatorFiltererUpgradeable.sol
OperatorFiltererUpgradeable.sol	OperatorFiltererUpgradeable.sol

1.5 Summary of findings

Severity	# of Findings
Critical	0
High	4
Medium	13
Low	17

ID	Name	Severity	Status
H-1	Contract works with non-existent collections and tokens	High	Fixed
H-2	Undefined behavior for `mint`	High	Fixed
H-3	An intruder can block any users	High	Fixed
H-4	`FantiumNFTV1. owners` shadows `ERC721Upgradeable._owners`	High	Fixed
M-1	Insufficient role isolation	Medium	Fixed
M-2	Insufficient constraint checks	Medium	Acknowledged
M-3	DefaultOperatorFiltererUpgradeable initializer is not called	Medium	Fixed
M-4	Unsafe Math	Medium	Fixed
M-5	Use `transfer` instead of `call`	Medium	Fixed
M-6	Broken `operatorFilterRegistry` may lead to DOS	Medium	Acknowledged
M-7	Wrong conditions in `mint`	Medium	Fixed
M-8	Multiple issues when migrating to new fantiumNFTContractAddress	Medium	Fixed
M-9	maxInvocation limits in two contracts are not synchronized	Medium	Fixed
M-10	Likely mistake in roles allowed to mint()	Medium	Fixed
M-11	Frontrun can create alternative NFT markets when price goes up	Medium	Acknowledged
M-12	Every `priceInWei` in collections requires updating after `updatePaymentToken`	Medium	Fixed
M-13	Use general safeTransferFrom	Medium	Acknowledged

L-1	Support additional EIPs	Low	Fixed
L-2	No additional information via events	Low	Acknowledged
L-3	Duplicate variables	Low	Fixed
L-4	Duplicate assignments	Low	Fixed
L-5	An athlete cannot update their address	Low	Acknowledged
L-6	Variables not used	Low	Acknowledged
L-7	ERC-1155 multi-token standard is not used	Low	Acknowledged
L-8	`operatorFilterRegistry` cannot be updated	Low	Acknowledged
L-9	Unsynchronized roles in two contracts, likely not designed	Low	Fixed
L-10	NFT Pause is limited	Low	Fixed
L-11	Tier information upgrading flow likely not designed	Low	Fixed
L-12	Current collectionIdToAllowList likely has practical limits	Low	Fixed
L-13	The addCollection() function can be made external	Low	Fixed
L-14	The field baseURI might not be initialized	Low	Acknowledged
L-15	Redundant check for zero address in the mintTo() function	Low	Fixed
L-16	New allocation mechanics for `allowList` can be bypassed when allocation is updated	Low	Fixed
L-17	Optimize gas	Low	Acknowledged

1.6 Conclusion

During the audit process, the developers spotted and acknowledged 4 HIGH, 13 MEDIUM, and 17 LOW severity findings. After working on the reported findings, all of them were acknowledged or fixed by the client.

File name	Contract deployed on mainnet
FantiumNFT	0x3fd236481d02C90C093f90792539318DE2b0007D
ERC1967Proxy (out of scope)	0x2b98132E7cfd88C5D854d64f436372838A9BA49d

2.FINDINGS REPORT

2.1 Critical

Not Found

2.2 High

H-1	Contract works with non-existent collections and tokens
Files	FantiumNFTV1.sol#L308 FantiumMinterV1.sol#L241
Severity	High
Status	Fixed in fdb089fa

Description

1. View methods (`getRoyalties()`, `tokenURI()`) work with non-existent collections.

In this example, incorrect royalty is returned for a non-existent collection:

```
await nftContract.getRoyalties(0);
# returns:
recipients: [
    '0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC',
    '0x0000000000000000000000000000000000000000'
]
bps: [
    BigNumber { value: "250" },
    BigNumber { value: "0" }
]
```

2. Athletes and Platform Manager can change parameters for non-existent collections. For example, function `updateCollectionAthleteAddress` (FantiumNFTV1.sol#L308).

3. A user can try to mint non-existent collections for free via `FantiumMinterV1.mint()` if `maxInvocations` is set. (FantiumMinterV1.sol#L241)

Recommendation

We recommend checking the existing `collectionId` and `tokenId`. An example of classical implementation:

```
require(_exists(tokenId), "Nonexistent token");
# or
function tokenURI(uint256 tokenId) public view virtual override
returns (string memory) {
    _requireMinted(tokenId);
    ...
}
```

H-2	Undefined behavior for `mint`
File	FantiumMinterV1.sol#L269
Severity	High
Status	Fixed in fdb089fa

Description

In `FantiumMinterV1.mint()`, param `_to` is intended to be `_to Address to be the minted token's owner`. It's wrong because the final mint will be for `msg.sender` (FantiumMinterV1.sol#L269).

```
# athlete -> 0x90F79bf6EB2c4f870365E785982E1f101E93b906
# fan -> 0x15d34AAf54267DB7D7c367839AAf71A00a2C6A65
await minterContract.connect(fan).mint(athlete.address, 1,
{ value: 100000000000000 });
console.log("ownerOf -> ", await nftContract.ownerOf(1000000));
# print fan -> 0x15d34AAf54267DB7D7c367839AAf71A00a2C6A65
```

Recommendation

We recommend changing it: `fantiumNFTContract.mintTo(_to, thisTokenId);`

H-3	An intruder can block any users
File	FantiumMinterV1.sol#L266
Severity	High
Status	Fixed in fdb089fa

Description

`FantiumMinterV1.mint()` allows passing `_to`, which is not validated.

After this line (FantiumMinterV1.sol#L266), we can change the permit of some users. In a common way we need to have the `onlyPlatformManager` permission for this action.

Recommendation

We recommend that you rethink the logic of `allowList`.

H-4	`FantiumNFTV1._owners` shadows `ERC721Upgradeable._owners`
File	FantiumNFTV1.sol#L245-L248
Severity	High
Status	Fixed in fdb089fa

Description

FantiumNFTV1 `mapping(uint256 => address) internal _owners` shadows ERC721Upgradeable `mapping(uint256 => address) private _owners`.

Thus, the `exists()` (FantiumNFTV1.sol#L245-L248) function will not work correctly:

```
function exists(uint256 _tokenId) public view returns (bool) {
    return _owners[_tokenId] != address(0);
}
```

Recommendation

1. Remove the FantiumNFTV1 `mapping(uint256 => address) internal _owners` line.
2. Use the `ERC721Upgradeable.ownerOf()` method.

2.3 Medium

M-1	Insufficient role isolation
Files	FantiumNFTV1.sol#L121 FantiumMinterV1.sol#L61
Severity	Medium
Status	Fixed in fdb089fa

Description

Roles are not sufficiently isolated:

1. `DEFAULT_ADMIN_ROLE` is allowed to run any operations on the contracts.
2. By default, this and other roles are all assigned to a deployer.

This increases the surface for social engineering attacks.

Recommendation

`DEFAULT_ADMIN_ROLE` should be used to grant roles only and nothing else. The deployer may have this role, but they should not be granted any other roles.

- FantiumNFTV1.sol#L121,
- FantiumNFTV1.sol#L130,
- FantiumMinterV1.sol#L61,
- FantiumMinterV1.sol#L71,
- FantiumMinterV1.sol#L211,
- FantiumMinterV1.sol#L221

It creates confusion between a role that is supposed to manage the technical side of the contract and the role of the platform manager responsible for the business logic consistency. This creates difficulties in role management and has the potential to introduce errors related to the functionality permitted to each role.

Thus, it is recommended to remove these lines in FantiumNFTV1.sol#L18:

```
_grantRole(PLATFORM_MANAGER_ROLE, msg.sender);
...
_grantRole(UPGRADER_ROLE, msg.sender);
```

```
...
hasRole(DEFAULT_ADMIN_ROLE, msg.sender),
...
hasRole(DEFAULT_ADMIN_ROLE, msg.sender),
```

Also, remove the lines in FantiumMinterV1.sol#L96-L98:

```
hasRole(DEFAULT_ADMIN_ROLE, msg.sender),
...
hasRole(DEFAULT_ADMIN_ROLE, msg.sender),
...
_grantRole(KYC_MANAGER_ROLE, msg.sender);
_grantRole(PLATFORM_MANAGER_ROLE, msg.sender);
_grantRole(UPGRADER_ROLE, msg.sender);
```

M-2	Insufficient constraint checks
Files	FantiumNFTV1.sol#L266 FantiumMinterV1.sol#L322 FantiumNFTV1.sol#L453-L456
Severity	Medium
Status	Acknowledged

Description

Insufficient constraint checks increase the human error probability: a platform manager may enter incorrect data and break the contract logic.

`addCollection()` (FantiumNFTV1.sol#L266):

- no zero address check for `athleteAddress`
- no constraint checks for `athletePrimarySalesPercentage`, `athleteSecondarySalesPercentage`

No check for the `collectionId` existence:

- `updateCollectionName` (FantiumNFTV1.sol#L297)
- `updateCollectionAthleteAddress` (FantiumNFTV1.sol#L308)
- `toggleCollectionIsPaused` (FantiumNFTV1.sol#L319)
- `updateCollectionPrice` (FantiumNFTV1.sol#L330)
- `updateCollectionMaxInvocations` (FantiumNFTV1.sol#L342)
- `updateCollectionTier` (FantiumNFTV1.sol#L353)
- `updateCollectionBaseURI` (FantiumNFTV1.sol#L366)
- `updateCollectionAthleteName` (FantiumNFTV1.sol#L378)
- `updateCollectionAthletePrimaryMarketRoyaltyPercentage` (FantiumNFTV1.sol#L397)
- `updateCollectionAthleteSecondaryMarketRoyaltyPercentage` (FantiumNFTV1.sol#L422)

No check `maxInvocations < ONE_MILLION` check:

- `updateCollectionMaxInvocations` (FantiumNFTV1.sol#L342)

No zero address check:

- `updateCollectionAthleteAddress` (FantiumNFTV1.sol#L308)
- `updateFantiumPrimarySaleAddress` (FantiumNFTV1.sol#L467)
- `updateFantiumSecondarySaleAddress` (FantiumNFTV1.sol#L479)

- `updateFantiumMinterAddress` (FantiumNFTV1.sol#L508)
- `updateFantiumNFTAddress` (FantiumMinterV1.sol#L322)

`updateTiers` (FantiumNFTV1.sol#L493):

- no check for `_priceInWei > 0`
- no check for `_maxInvocations < ONE_MILLION`

`fantiumPrimarySalesAddress` in `getPrimaryRevenueSplits()` (FantiumNFTV1.sol#L600) may be zero. Consider setting it to a non-zero address in the contract's initializer so it is never zero.

Recommendation

It is recommended to add all necessary checks.

Client's commentary

> Client: Partially fixed in commit fdb089fa02c36560645f5ae7a3ab06d63f37ee1f
> Mixbytes: No constraint checks at FantiumNFTV1.sol#L453-L456

M-3	DefaultOperatorFiltererUpgradeable initializer is not called
File	FantiumNFTV1.sol#L168
Severity	Medium
Status	Fixed in fdb089fa

Description

FantiumNFTV1 class inherits *DefaultOperatorFiltererUpgradeable* but it's initializer is not called together with other base classes initializers in the `initialize()` method at FantiumNFTV1.sol#L168.
Due to this, the full functionality related to OperatorFilterRegistry becomes unavailable.

Recommendation

We recommend adding the `DefaultOperatorFilterer_init()` call to *FantiumNFTV1* initializer.

M-4	Unsafe Math
File	FantiumNFTV1.sol#L620
Severity	Medium
Status	Fixed in fdb089fa

Description

In the project, unvalidated params with unsafe math are used. For example, it's dangerous logic (FantiumNFTV1.sol#L620):

```
unchecked {
    // fantiumRevenue_ is always <=25, so guaranteed to never underflow
    collectionFunds = _price - athleteRevenue_;
}
```

In this case, `collection.athletePrimarySalesPercentage`, `_price`, `_collectionId` are not validated fully.

Recommendation

We recommend using only safe math for all operations in the project.

M-5	Use `transfer` instead of `call`
File	
Severity	Medium
Status	Fixed in fdb089fa

Description

If the project work with other contracts is not implied, then the `transfer` method is safer (this forwards 2300 gas).

Recommendation

We recommend using `transfer` to avoid potential re-entrancy.

Client's commentary

> Mixbytes: To avoid re-entrancy ERC777 should be discarded.

M-6	Broken `operatorFilterRegistry` may lead to DOS
File	OperatorFiltererUpgradeable.sol#L10
Severity	Medium
Status	Acknowledged

Description

If the `operatorFilterRegistry` (OperatorFiltererUpgradeable.sol#L10) is broken and reverts on every transaction, then all transfers with `modifier onlyAllowedOperator` (OperatorFiltererUpgradeable.sol#L32) will be locked.

Recommendation

It is recommended to allow the platform manager to update `operatorFilterRegistry` (OperatorFiltererUpgradeable.sol#L10).

Client's commentary

> OpenSea has now implemented their own upgradeable contract which we are using now. We will override via upgrades in case of issues.

M-7	Wrong conditions in `mint`
File	FantiumMinterV1.sol#L200
Severity	Medium
Status	Fixed in fdb089fa

Description

Wrong conditions in `mint` (FantiumMinterV1.sol#L200):

```
if (
    !hasRole(PLATFORM_MANAGER_ROLE, msg.sender) ||
    !hasRole(DEFAULT_ADMIN_ROLE, msg.sender)
) {
    require(isAddressKYCed(msg.sender), "Address not KYCed");
}
IFantiumNFT.Collection memory collection = fantiumNFTContract
    .getCollection(_collectionId);
// sender must be on allow list or Admin or Manager if the collection is pause
if (
    !hasRole(PLATFORM_MANAGER_ROLE, msg.sender) &&
    !hasRole(DEFAULT_ADMIN_ROLE, msg.sender)
) {
    require(
        !collection.paused ||
            isAddressOnAllowList(_collectionId, msg.sender),
        "Purchases are paused and not on allow list"
    );
}
```

Recommendation

The correct (but not the optimal) form is to use the "&&" operator:

```
if (
    !hasRole(PLATFORM_MANAGER_ROLE, msg.sender) &&
    !hasRole(DEFAULT_ADMIN_ROLE, msg.sender)
)
```

But the best way would be to restrict privileges and disallow unnecessary access altogether:

```
require(isAddressKYCed(msg.sender), "Address not KYCed");
IFantiumNFT.Collection memory collection = fantiumNFTContract
    .getCollection(_collectionId);

require(!collection.paused, "Purchases are paused");
```

M-8	Multiple issues when migrating to new fantiumNFTContractAddress
File	FantiumMinterV1.sol#L322-L329
Severity	Medium
Status	Fixed in fdb089fa

Description

FantiumMinterV1 allows setting a new fantiumNFTContractAddress with updateFantiumNFTAddress(). It only changes the address stored but does not affect other scenarios.

1. The new fantiumNFTContractAddress will have an empty `tokenId` and `collections` sold, allowing minting NFTs with the same `tokenId`.
2. Mapping `collectionIdToAllowList` remains unchanged, thus the old allowlist for old collections will be active for new collections.
3. Minting on the old contract will be frozen.

When a new fantiumNFTContractAddress is set, it will be possible.
When `FantiumMinterV1` changes its `fantiumNFTContractAddress`, its stored mapping `collectionIdToAllowList` remains unchanged.

- FantiumMinterV1.sol#L322-L329

Recommendation

The general recommendation is to update code on scenarios of migration to the new FantiumNFTV1. We assume some possible steps are as follows:

- store the allowlist at `FantiumNFTV1` only
- remove migration functionality at all

M-9	maxInvocation limits in two contracts are not synchronized
File	FantiumNFTV1.sol#L346
Severity	Medium
Status	Fixed in fdb089fa

Description

FantiumMinterV1 imply that maxInvocation for collection is limited by 1 mln. But multiple setter functions in FantiumNFTV1 allow maxinvocation for collections being above 1 mln.

- FantiumNFTV1.sol#L346
- FantiumNFTV1.sol#L493-L505

Recommendation

For every function that modifies `maxInvocation`, we recommend checking that it is limited by 1 mln.

M-10	Likely mistake in roles allowed to mint()
File	FantiumMinterV1.sol#L210-L230
Severity	Medium
Status	Fixed in fdb089fa

Description

At FantiumMinterV1 function mint() has the following lines:

```
// sender must be KYCed or Admin or Manager
        if (
            !hasRole(PLATFORM_MANAGER_ROLE, msg.sender) ||
            !hasRole(DEFAULT_ADMIN_ROLE, msg.sender)
        ) {
            require(isAddressKYCed(msg.sender), "Address not KYCed");
        }
```

msg.sender is required to be KYCed, except for the case when it has both a PlatfromManager and DefaultManager role.
But the comment states that `sender must be KYCed or Admin or Manager`.
Thus, it is not expected in this function that it will revert in case when PlatformManager calls without the DefaultAdmin role (or DefaultAdmin without the PlatformManager role).

- FantiumMinterV1.sol#L210-L230

Recommendation

As the expected behavior is likely written in the comments, we recommend changing `||` to `&&` in the line above. It will correlate with the following lines of code where the same requires checking with `&&`.

M-11	Frontrun can create alternative NFT markets when price goes up
File	
Severity	Medium
Status	Acknowledged

Description

FantiumNFTV1 allows upgrading the price of NFTs. When the price for a collection is set higher by calling updateCollectionPrice(), an attacker can front-run this price increase, buy many NFT tokens, and arrange secondary markets for these tokens. The price can be lower than the one set after updateCollectionPrice().

Recommendation

The issue is hard to mitigate. One of the options is to introduce workflow when a caller pauses minting before calling updateCollectionPrice().

Client's commentary

> Mixbytes: not fixed as the new merged contract allows price updating with updateTiers() or updateCollectionTier().

M-12	Every `priceInWei` in collections requires updating after `updatePaymentToken`
Files	e79ed7dd FantiumNFTV1.sol#L717
Severity	Medium
Status	Fixed in 4307c73d

Description

In commit e79ed7dd a new method will update the payment method (FantiumNFTV1.sol#L717).

```
function updatePaymentToken(
    address _address
) external onlyRole(PLATFORM_MANAGER_ROLE) {
    require(_address != address(0));
    erc20PaymentToken = _address;
}
```

Tokens have different decimals, so after calling `updatePaymentToken` it will be necessary to update the price for each collection.

Recommendation

We recommend updating `erc20PaymentToken` to check that the decimals have not changed. Otherwise, you will have to update each price as it corresponds to the previous token.

M-13	Use general safeTransferFrom
File	FantiumNFTV1.sol#L372
Severity	Medium
Status	Acknowledged

Description

In commit e79ed7dddabc482c56f7828bd9a8725fbbeca2f5, it is required to check the success of the transfer at the following lines:

- FantiumNFTV1.sol#L372
- FantiumNFTV1.sol#L380.

Recommendation

It is recommended to always use the `safeTransferFrom()` function when sending tokens.

2.4 Low

L-1	Support additional EIPs
File	
Severity	Low
Status	Fixed in fdb089fa

Description

The project uses `EIP2981`. For example, `FantiumNFTV1` cannot support this EIP via the `supportsInterface` view method.

```
# In supportsInterface
interfaceId == type(IEIP2981).interfaceId
interfaceId == type(IEIP2981RoyaltyOverride).interfaceId
```

Recommendation

We recommend adding supported EIPs in the `supportsInterface` method if needed.

L-2	No additional information via events
File	FantiumNFTV1.sol#L90
Severity	Low
Status	Acknowledged

Description

These events do not contain all information about the update:

- `PlatformUpdated` (FantiumNFTV1.sol#L90)
- `CollectionUpdated` (FantiumNFTV1.sol#L86)

And for method there is no event:

- `updateTiers` FantiumNFTV1.sol#L493

Recommendation

We recommend adding more event info for user convenience.

L-3	Duplicate variables
File	
Severity	Low
Status	Fixed in fdb089fa

Description

The FantiumMinterV1 variables `fantiumNFTContract` and `fantiumNFTContractAddress` duplicate each other.

Recommendation

It is recommended to remove either `fantiumNFTContract` or `fantiumNFTContractAddress` to save gas.

L-4	Duplicate assignments
File	FantiumNFTV1.sol#L281
Severity	Low
Status	Fixed in fdb089fa

Description

There are duplicate lines:

- FantiumNFTV1.sol#L281
- FantiumNFTV1.sol#L283

Recommendation

It is recommended to remove one of the lines to save gas.

L-5	An athlete cannot update their address
File	FantiumNFTV1.sol#L308
Severity	Low
Status	Acknowledged

Description

An athlete cannot update their address in `updateCollectionAthleteAddress()` (FantiumNFTV1.sol#L308).

Recommendation

It is recommended to check if this behavior is intended.

Client's commentary

> It's intended.

L-6	Variables not used
File	FantiumNFTV1.sol#L37
Severity	Low
Status	Acknowledged

Description

The FantiumNFTV1 contract does not use variables:

- `KYC_Manager`
- `FIELD_FANTIUM_PRIMARY_MARKET_ROYALTY_PERCENTAGE`
- `tierSet`

Lines:

- FantiumNFTV1.sol#L37
- FantiumNFTV1.sol#L45
- FantiumNFTV1.sol#L33

Recommendation

We recommend removing these variables.

Client's commentary

> Mixbytes: `FIELD_FANTIUM_PRIMARY_MARKET_ROYALTY_PERCENTAGE` and `fantiumMinterAddress` are not used.

L-7	ERC-1155 multi-token standard is not used
File	
Severity	Low
Status	Acknowledged

Description

ERC-1155 is a smart contract interface that can represent and control any number of fungible and non-fungible token types. In this way, the ERC-1155 token can do the same functions as an https://ethereum.org/en/developers/docs/standards/tokens/erc-20/ and https://ethereum.org/en/developers/docs/standards/tokens/erc-721/ token, and even both at the same time. And best of all, improving the functionality of both standards, making it more efficient, and correcting obvious implementation errors on the ERC-20 and ERC-721 standards.

The ERC-1155 token is described fully in https://eips.ethereum.org/EIPS/eip-1155.

Recommendation

It is recommended to take note of the https://eips.ethereum.org/EIPS/eip-1155 multi-token standard: it can be used for NFT collections.

Client's commentary

> Good points. We compared both standards in the beginning and decided for 721 based on requirements.

L-8	`operatorFilterRegistry` cannot be updated
File	OperatorFiltererUpgradeable.sol#L10
Severity	Low
Status	Acknowledged

Description

The `operatorFilterRegistry` (OperatorFiltererUpgradeable.sol#L10) constant cannot be updated:

```
IOperatorFilterRegistry constant operatorFilterRegistry =
    IOperatorFilterRegistry(0x000000000000AAeB6D7670E522A718067333cd4E);
```

Recommendation

It is recommended to set `operatorFilterRegistry` in the initializer or in the constructor and also add a method to update the variable.

The update method must allow setting the variable to zero: this allows replacing the `code.length` (OperatorFiltererUpgradeable.sol#L34) check in the `onlyAllowedOperator` modifier to a zero address check.

Client's commentary

> OpenSea has now implemented their own upgradeable contract which we are using now. We will override via upgrades in case of issues.

L-9	Unsynchronized roles in two contracts, likely not designed
Files	FantiumMinterV1.sol#L25-L28 FantiumNFTV1.sol#L37-L40
Severity	Low
Status	Fixed in fdb089fa

Description

Two contracts share the same roles and likely expect them to be the same.
`FantiumNFTV1` may expect that a function with modifiers onlyPlatformManager() and onlyAthlete() is allowed for the same `PLATFORM_MANAGER_ROLE`, as the `PLATFORM_MANAGER_ROLE` is allowed to mint NFT tokens - but this `PLATFORM_MANAGER_ROLE` is stored at FantiumMinterV1 and can be different.

- FantiumMinterV1.sol#L25-L28
- FantiumNFTV1.sol#L37-L40

Recommendation

If roles are expected to have the same addresses on both contracts, it is recommended to consider one single storage for roles OR remove doubled roles OR implement different role namings.

L-10	NFT Pause is limited
File	FantiumNFTV1.sol#L323
Severity	Low
Status	Fixed in fdb089fa

Description

The FantiumNFTV1 pause for collections blocks only new mintings. In addition, the address at allowedList still can mint.

Transfers remain allowed. So the scenario is possible when `toggleCollectionIsPaused()` is front run, and NFT can be traded on secondary markets as transfers are still allowed.

- FantiumNFTV1.sol#L323

Recommendation

Consider renaming the pause to `isMintAllowed` or add a transfer pausing if it is required by the pause design.

Client's commentary

> Mixbytes: two types of pauses are separated after the fix:
>
> 1. Global pause for the merged contract
> 2. `isMintingPaused` for a collection, setting `true` means that only the allowed list of addresses can mint NFTs based on their allocation.

L-11	Tier information upgrading flow likely not designed
File	FantiumNFTV1.sol#L493-L505
Severity	Low
Status	Fixed in fdb089fa

Description

We outline that updating Tiers in `tiers` with dedicated functions do not change tier information stored at Collections in `collections`.

In addition, collection parameters related to tiers can be easily changed and thus deviate from predefined Tier stores at `tiers`.

- FantiumNFTV1.sol#L493-L505

Recommendation

We recommend either removing Tiers at all or removing allowance to modify Tier information in collections.

Client's commentary

> Client: Fixed in commit fdb089fa02c36560645f5ae7a3ab06d63f37ee1f.
> Mixbytes: a collection now only stores a reference to a tier name, while tier params are stored in a separate mapping. Tier updating will change the data for all collections.

L-12	Current collectionIdToAllowList likely has practical limits
File	FantiumMinterV1.sol#L225-L266
Severity	Low
Status	Fixed in fdb089fa

Description

`allowList` is checked when `mint()` happens at `FantiumMinterV1`. It always sets false for an address (buyer). But if it is true, it allows only one mint for an address when a collection is paused. This behavior is strange and likely not to be designed.

- FantiumMinterV1.sol#L225-L266

Recommendation

We recommend designing a more clear flow for `collectionIdToAllowList` and its behavior during pauses. By the way, we assume that the current `collectionIdToAllowList` behaves as designed and no code modification is needed.

L-13	The addCollection() function can be made external
File	FantiumNFTV1.sol#L272
Severity	Low
Status	Fixed in fdb089fa

Description

The `addCollection()` function at (FantiumNFTV1.sol#L272) is defined as public. As the function is not used for calls from smart contracts, it can be made external to save gas.

Recommendation

We recommend making the `addCollection()` function external.

L-14	The field baseURI might not be initialized
File	FantiumNFTV1.sol#L236
Severity	Low
Status	Acknowledged

Description

The `tokenURI()` method uses the *baseURI* field at (FantiumNFTV1.sol#L236) which is meant to be initialized by the `updateBaseURI()` method. It is not verified whether the method was called during the contract initialization. This can result in an incorrect output of URI address in case collectionURI was not initialized.

Recommendation

We recommend checking whether the baseURI field was initialized before using it.

L-15	Redundant check for zero address in the mintTo() function
File	FantiumNFTV1.sol#L208
Severity	Low
Status	Fixed in fdb089fa

Description

The check is redundant as the check on the next line at FantiumNFTV1.sol#L208

```
require(fantiumMinterAddress != address(0), "Fantium Minter not set");
```

Recommendation

We recommend removing the redundant check to save gas.

L-16	New allocation mechanics for `allowList` can be bypassed when allocation is updated
File	fdb089fa
Severity	Low
Status	Fixed in e79ed7dd

Description

Fix in commit fdb089fa introduced a new data type for `allowList`. Now it is a mapping storing the allocation for a user in a collection:
`mapping(uint256 => mapping(address => uint256)) public collectionIdToAllowList`

Allocation is the number of tokens allowed to the mint for an address. Also, two new functions set these allocations:
`function addAddressToAllowListWithAllocation(uint256 _collectionId,address _address,uint256 _allocation`
`function reduceAllowListAllocation(uint256 _collectionId,address _address,bool _completely)`

These two functions can be bypassed in some scenarios:

1. `addAddressToAllowListWithAllocation()` is a setter that can be used to increase or decrease the allocation. When for example, the allocation is updated from 5 to 10, the address can front-run the update, buy 5 NFTs, wait for the update confirmation and buy an additional 10 NFTs as it is a newly updated allocation. Thus, the allocation update allows us to bypass of allocation limits and purchase additional NFTs.
2. The same thing for `reduceAllowListAllocation()`. But it can only be exploited when `_completely` is used. The impact here is limited as the worst scenario allows purchasing the previous allocation before the update.

In addition, `reduceAllowListAllocation()` decreases the allocation by one point per call. For instance, it can be costly to decrease the allocation from 20 to 5.

Recommendation

We recommend removing setters and implementing `increase`/`decrease` mechanics.

1. it is better if `addAddressToAllowListWithAllocation` increments the allocations like:

```
collectionIdToAllowList[_collectionId][_address] += _allocationAdded
```

2. it is better if `reduceAllowListAllocation` does the same thing when decreasing:

```
collectionIdToAllowList[_collectionId][_address] -= _allocationReduced
```

L-17	Optimize gas
Files	e79ed7dd FantiumNFTV1.sol#L316
Severity	Low
Status	Acknowledged

Description

In commit e79ed7dd the check is redundant:

- FantiumNFTV1.sol#L316

```
require(
    IERC20(erc20PaymentToken).allowance(msg.sender, address(this)) >=
        collection.priceInWei,
    "ERC20 allowance too low"
);
```

This check is optional, `transferFrom` will revert if approval is absent.

Recommendation

We recommend removing the redundant check to save gas.

2.5 Appendix

Monitoring Recommendation

The project contains smart contracts that require active monitoring. For these purposes, it is recommended to proceed with developing new monitoring events based on Forta (https://forta.org) with which you can track the following exemplary incidents:

- Someone is buying up NFT collections in bulk;
- The collection is full (due to `maxInvocations`);
- Transferring NFT is blocked by OpenSea Filter (`only Allowed Operator`).

3. ABOUT MIXBYTES

MixBytes is a team of blockchain developers, auditors and analysts keen on decentralized systems. We build opensource solutions, smart contracts and blockchain protocols, perform security audits, work on benchmarking and software testing solutions, do research and tech consultancy.

Contacts

https://github.com/mixbytes/audits_public

https://mixbytes.io/

hello@mixbytes.io

https://twitter.com/mixbytes