Competitive Security Assessment

FTM NFT

Nov 24th, 2022

Secure3

secure3.io

Summary

This report is prepared for the project to identify vulnerabilities and issues in the smart contract source code. A group of NDA covered experienced security experts have participated in the Secure3's Audit Contest to find vulnerabilities and optimizations. Secure3 team has participated in the contest process as well to provide extra auditing coverage and scrutiny of the finding submissions.

The comprehensive examination and auditing scope includes:

- Cross checking contract implementation against functionalities described in the documents and white paper disclosed by the project owner.
- Contract Privilege Role Review to provide more clarity on smart contract roles and privilege.
- Using static analysis tools to analyze smart contracts against common known vulnerabilities patterns.
- Verify the code base is compliant with the most up-to-date industry standards and security best practices.
- Comprehensive line-by-line manual code review of the entire codebase by industry experts.

The security assessment resulted in findings that are categorized in four severity levels: Critical, Medium, Low, Informational. For each of the findings, the report has included recommendations of fix or mitigation for security and best practices.

Overview

Project Detail

Project Name	FTM NFT
Platform & Language	Move / Aptos
Codebase	• https://github.com/FTM-Labs/nft-contracts • audit commit - 77a0c6d24e6e86c7652f02fa51667fa15151b875 • final commit - 3872d939264fe077db5e0742258cfdd81a145e76
Audit Methodology	• Audit Contest • Business Logic and Code Review • Privileged Roles Review • Static Analysis

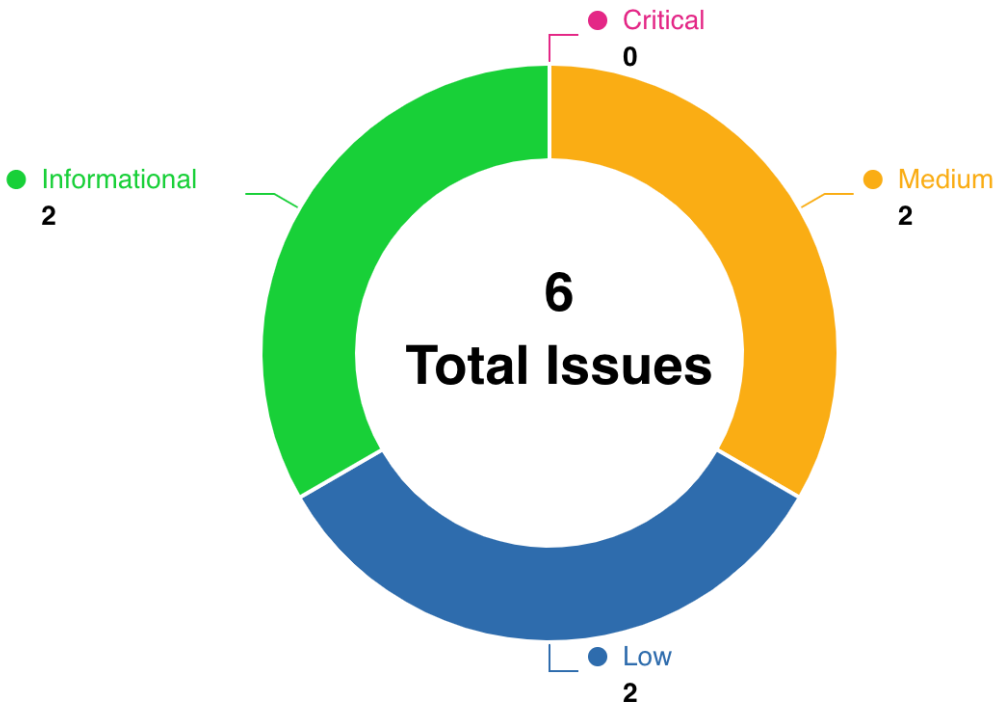
Code Vulnerability Review Summary

Vulnerability Level	Total	Reported	Acknowledged	Fixed	Mitigated	Declined
Critical	0	0	0	0	0	0
Medium	2	0	0	2	0	0
Low	2	0	0	2	0	0
Informational	2	0	0	2	0	0

Audit Scope

File	Commit Hash
candy_machine_v2.move	77a0c6d24e6e86c7652f02fa51667fa15151b875

Code Assessment Findings



ID	Name	Category	Severity	Status	Contributor
FTM-1	Duplicate check for array length	Gas Optimization	Informational	Fixed	Secure3
FTM-2	Missing function to mutate token settings	Logical	Low	Fixed	Secure3
FTM-3	The used seed will cause the function `create_cm_v2` to fail to run	Logical	Medium	Fixed	Secure3
FTM-4	`CollectionConfig.is_public` is set but never used	Logical	Low	Fixed	Secure3
FTM-5	`exists` check after `move_to` is unnecessary	Gas Optimization	Informational	Fixed	Secure3
FTM-6	`mint_tokens` incorrect mint number check	Logical	Medium	Fixed	Secure3

FTM-1:Duplicate check for array length

Category	Severity	Code Reference	Status	Contributor
Gas Optimization	Informational	• code/nft-contracts/sources/candy_machine_v2.move#L242-L244	Fixed	Secure3

Code

```
242:        assert!(vector::length(&token_names) == vector::length(&property_keys),
ETOKEN_INFO_SIZE_NOT_EQ);
243:        assert!(vector::length(&token_names) == vector::length(&property_values),
ETOKEN_INFO_SIZE_NOT_EQ);
244:        assert!(vector::length(&token_names) == vector::length(&property_types),
ETOKEN_INFO_SIZE_NOT_EQ);
```

Description

Secure3 : In the function `upload_nft()`, the length of the array parameters is repeatedly(#L238-L240) checked

Recommendation

Secure3 : Delete the above checks to save gas.

Client Response

Fixed

FTM-2:Missing function to mutate token settings

Category	Severity	Code Reference	Status	Contributor
Logical	Low	• code/nft-contracts/sources/candy_machine_v2.move#L248	Fixed	Secure3

Code

```
248:         let token_mutate_config = token::create_token_mutability_config(&mutate_setting);
```

Description

Secure3 : `admin` can decide whether the token settings can be mutated when uploading the token. But there is no function for `admin` to mutate token settings.

Recommendation

Secure3 : Add a function that allows `admin` to mutate the token settings.

Client Response

Fixed

FTM-3:The used seed will cause the function `create_cm_v2` to fail to run

Category	Severity	Code Reference	Status	Contributor
Logical	Medium	• code/nft-contracts/sources/candy_machine_v2.move#L54-L61	Fixed	Secure3

Code

```
54:    public entry fun create_cm_v2(admin: &signer) {
55:        let admin_addr = signer::address_of(admin);
56:
57:        // TODO: check how to create resouce account with better seed.
58:        let (_, signer_cap) = account::create_resource_account(admin, x"01");
59:        move_to(admin, ResourceData { resource_account: signer_cap });
60:        assert!(exists(admin_addr), ERESOURCE_ACCOUNT_NOT_CREATED);
61:    }
```

Description

Secure3 : The function `create_cm_v2` uses the fixed seed `x"01"` to create the signer's resource account. If the seed `x"01"` is already used by another project, the function `create_cm_v2` will always revert because its internal call `account::create_resource_account` cannot create a resource account with a used seed.

Recommendation

Secure3 : Consider generating a random seed from the timestamp to ensure that the seed is not used.
Consider below fix in the `create_cm_v2` function

```
public entry fun create_cm_v2(admin: &signer) {
// ...
let seed = utf8_utils::u128_to_string((timestamp::now_microseconds() as u128));
string::append(&mut seed, string::utf8(b"candy_machine_v2"));
let (_, signer_cap) = account::create_resource_account(admin, *string::bytes(&seed));
// ...
}
```

You might also consider generating a random seed off-chain, and pass it as an argument to this function.

Client Response

Fixed

FTM-4: `CollectionConfig.is_public` is set but never used

Category	Severity	Code Reference	Status	Contributor
Logical	Low	• code/nft-contracts/sources/candy_machine_v2.move#L38-L40	Fixed	Secure3

Code

```
38:     struct CollectionConfig has key, store {
39:         admin: address,
40:         is_public: bool,
```

Description

Secure3 : `is_public` is one of the properties of `CollectionConfig` and it can be modified by admin. But `is_public` is not used in other parts of the code.

Recommendation

Secure3 : Check the effect of `is_public` in the code. If `is_public` is a switch variable for permission management, add corresponding code logic to `upload_nft` function or `mint_tokens` function.

Client Response

Fixed

FTM-5: `exists` check after `move_to` is unnecessary

Category	Severity	Code Reference	Status	Contributor
Gas Optimization	Informational	• code/nft-contracts/sources/candy_machine_v2.move#L59-L60 • code/nft-contracts/sources/candy_machine_v2.move#L366-L371	Fixed	Secure3

Code

```
59:         move_to(admin, ResourceData { resource_account: signer_cap });
60:         assert!(exists(admin_addr), ERESOURCE_ACCOUNT_NOT_CREATED);

366:             move_to(
367:                 creator,
368:                 CollectionConfigs { collection_configs: table::new() } );
369:          assert!(
370:             exists(addr),
371:             ECOLLECTIONS_CONFIG_NOT_CREATED);
```

Description

Secure3 : It is not possible that `exists` statement returns `false` after `move_to` statement, so the above codes can be removed to save gas.

Recommendation

Secure3 : Delete the above checks to save gas.

Client Response

Fixed

FTM-6: `mint_tokens` incorrect mint number check

Category	Severity	Code Reference	Status	Contributor
Logical	Medium	• code/nft-contracts/sources/candy_machine_v2.move#L306-L308	Fixed	Secure3

Code

```
306:         assert!(
307:             *minted <= vector::length(all_token_data), NO_MORE_NFT
308:         );
```

Description

Secure3 : Function `mint_tokens()` checks if `*minted` is less than the length of available tokens vector. But according to the code logic, `*minted` represents the number of tokens the user has minted.

Consider the following situation, the user has minted two tokens, and wants to mint the third token in this call, and there is only one left in the vector. Since `2>1` , the user will not be able to complete the minting.

Here may be to check `amount` instead of `*minted` to ensure that the number of remaining tokens is greater than or equal to the number that the user wants to mint in this call.

Recommendation

Secure3 : Consider below fix in the `mint_tokens()` function

```
assert!(
    amount <= vector::length(all_token_data), NO_MORE_NFT
);
```

Client Response

Fixed

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 Invoices, or the scope of services, and terms and conditions provided to you (“Customer” or the “Company”) in connection with the Invoice. This report provided in connection with the services set forth in the Invoices shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Invoice. 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 Secure3’s prior written consent in each instance.

This report is not an “endorsement” or “disapproval” of any particular project or team. This report is not an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Secure3 to perform a security assessment. This report does not provide any warranty or guarantee of free of bug of codes analyzed, nor do they provide any indication of the technologies, business model or legal compliancy.

This report should not be used in any way to make decisions around investment or involvement with any particular project. Instead, it represents an extensive assessing process intending to help our customers increase the quality of their code and high-level consistency of implementation and business model, while reducing the risk presented by cryptographic tokens and blockchain technology.

Secure3’s position on the final decisions over blockchain technologies and corresponding associated transactions is that each company and individual are responsible for their own due diligence and continuous security.

The assessment services provided by Secure3 is subject to dependencies and under continuing development. 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.