Canonical LLC

Security Assessment

Artifct

© 2022 Canonical LLC

Summary

This verification report is for the Artifct NFT purchase smart contract.

Through a process of manual review and white box testing, eight issues were found. Zero critical issues were found.

The audit was focused on finding security vulnerabilities. Special consideration was paid to adversarial usage of the contract that could result in lost assets for users.

The report also include informational issues, we believe are general improvements, but they are not relevant from a security perspective.

Issues

ID	Title	Severity	Status
1	A “double satisfaction” vulnerability.	Major	Resolved by 5442875
2	Possible missing assets when purchasing.	Major	Resolved by 5442875
3	Seller can receive 1 Ada less than the expected amount.	Major	Resolved in 6b33ae8
4	Price is not enforced with reused royalty address.	Major	Resolved in 91b7332
5	Price is not enforced with reused marketplace address.	Minor	Responded
6	Update code path is unnecessary.	Informational	Resolved by 6b33ae8
7	Unnecessary ShopDatum wrapper.	Informational	Resolved by cbc6a31
8	`checkN` prevents valid transactions.	Informational	Resolved by 6b33ae8

Background

Artifct is in development NFT marketplace. Artifct's buy now contract is designed to facilitate the buying and selling of NFTs.

Users first send an NFT along with a matching datum hash to the script address. The datum encodes, the seller's public key hash, the price, asset information, and royalty info. When a buyer attempts to unlock the NFT, the buyer must pay the seller, the royalty address, and the marketplace address. The amounts the buyer must pay are determined by a hard-coded marketplace percentage, and a dynamic royalty percentage.

Additionally, the contract allows the seller to regain control of the locked assets at any point through a "Cancel" redeemer. Moreover, the smart contract has a specialized code path for updating the price using the "Update" redeemer.

The contract is written to be used with NFTs. However, due to how NFTs are implemented in Cardano, it can be used with any non-native token. This includes tokens with multiplicity greater than one.

Information

Repo	https://github.com/Sbcdn/artifct-lbuc-sc
Commit	4f3cc33da170dec734526f5523823eed9259df2f
Address	addr1w9andn8q8vckn6j69nh7u52qqgxjhrmgtc2d22xgkl5wgxs95253k

1. A “double satisfaction” vulnerability.

Severity	Status
Major	Resolved by 5442875

Description

The smart contract attempts to prevent the "double satisfaction" attack by verifying there is only one entry in the datum list of the script context.

```
oneScript :: [(DatumHash, Datum)] -> Bool
oneScript l = length l == 1
```

In general, this approach works, but fails if multiple script inputs have identical datum hashes.

This implies the same NFT is listed twice. This might seem impossible, but it is important to remember that Cardano "NFTs" are merely non-native tokens with special metadata. It is not unheard of for "NFTs" listed on marketplaces to be fungible tokens with NFT metadata.

If these tokens with multiplicity over one were listed, they are vulnerable to "double satisfaction" attack.

Recommendation

Instead of verifying that there is only one datum, check that there is only one script address as an input.

2. Possible missing assets when purchasing.

Severity	Status
Major	Resolved by 5442875

Description

The smart contract ensures that the buyer receives a token. However, it does not verify that the buyer received a token from the script address.

This opens the door for an attacker to not lock any assets, but create a datum hash that implies it has. If the buyer has the exact token in their wallet, as listed in the datum, the purchase transaction can succeed.

This will result in buyer losing all funds and not receiving an asset.

Recommendation

Verify that the script is holding the proper value using `findOwnInput` or a similar approach.

3. Seller price 1 Ada too low with zero percent royalties

Severity	Status
Major	Resolved in 6b33ae8

Description

If the royalty amount is set to zero, the smart contract will validate even if no assets are paid to the royalty address, as specified by the following code:

```
sRr <= 0 || isPaid sR roy
```

However, the royalty amount, **`roy`**, will still evaluate to the **`minAda`** amount, e.g. 1 Ada.

This amount will be subtracted from the amount the seller should receive:

```
price :: Integer
price = sPrice - fee' - roy
```

Which will result in a `price` that is one Ada too small; thus a buyer could purchase the token for 1 Ada too little.

Recommendation

Only subtract the royalty value if it is not set to zero.

```
price :: Integer
price = sPrice - fee' - if sRr == 0 then 0 else roy
```

4. Price is not enforced with reused royalty address

Severity	Status
Major	Reported

Description

If the seller address and the royalty address are the same, the checks `isPaid sR roy` and `isPaid sSeller price` can pass when the amount paid equals `max roy price`. However, the transaction should ensure the amount paid is `roy + price`.

This can result in a seller not receiving all the funds they are entitled to.

Recommendation

Add a check for it the addresses are the same, and in that case, ensure the paid out value is the sum `roy + price`.

5. Price is not enforced with reused marketplace address

Severity	Status
Minor	Reported

Description

This is similar to issue **4**, but with the marketplace address instead of the royalty address. If the marketplace address and the seller address are the same, the payment checks will pass as long as **`max fee' price`** is paid to the marketplace/seller address. However, **`fee' + price`** should be paid.

Recommendation

Add a check for it the addresses are the same, and in that case, ensure the paid out value is the sum **`fee' + price`**.

Artifct Response

Issue number five is considered minor severity and just affects the wallet holding the marketplace fees and affects no marketplace users.

The marketplace fees wallet shall not be used to list tokens on the marketplace as the smart contract does not ensure the marketplace fee is paid for this specific address. However, this is a very rare case just concerning one wallet address. Artifct decided to not implement a mechanism to protect against this case as it would increase fees for all users with no change to security for users.

6. Update Code Path is Unnecessary

Severity	Status
Informational	Resolved by 6b33ae8

Description

In addition to `Buy` and `Cancel` the smart contract includes an `Update` code path. This is used to update information for a listing.

Here is the code:

```
validateCancel ::  NftShop -> ATxInfo -> Bool
validateCancel NftShop{..} info
    | txSignedBy' (atxInfoSignatories info) sSeller = True
    | otherwise = False
```

However, since the user signed the transaction, they are also responsible for setting the datum hash. Therefore, the user knows if the datum hash is valid, and passing the hash in two places to ensure they are the same does not enhance the security.

Recommendation

Remove the `Update` redeemer constructor and update code path. Instead of having a special update path that does not add any additional security, use the `Cancel` operation for updating.

7. Unnecessary ShopDatum Wrapper

Severity	Status
Informational	Resolved by 6b33ae8

Description

The datum has the following type:

Here is the code:

```
data ShopDatum = Shop NftShop
```

The additional wrapping of **`NftShop`** serves no purpose. The datum could just be **`NftShop`.**

Recommendation

Use **`NftShop`** as the datum and remove **`ShopDatum`**.

8. `checkN` Prevents Valid Transactions

Severity	Status
Informational	Resolved by 6b33ae8

Description

The main function to ensure a buyer receives the correct NFT is `checkN`:

```
checkN :: ATxInfo -> PubKeyHash -> CurrencySymbol -> TokenName -> Bool
checkN i p c t =
  let oneToken =
        PlutusTx.Prelude.filter
          (\(c',t',i') -> c' == c && t' == t && i' == 1)
          (flattenValue (valuePaidTo' i p))
  in length oneToken == 1
```

This implementation ensures a buyer only receives a single purchased token.

However, if the token is not actually an NFT, the buyer could have more in their wallet that are added as outputs to the transaction. If this were to occur, they would not be able to purchase the token.

Recommendation

Use `valueOf` to ensure there is at least one token:

```
checkN :: ATxInfo -> PubKeyHash -> CurrencySymbol -> TokenName -> Bool
checkN i p c t = valueOf (valuePaidTo' i p) c t > 1
```

Disclaimer

This report should is not an endorsement of Artifct or the smart contract being audited. The report should not be used to make investments or other financial decisions.

Using the Cardano blockchain entails risk and the user of any smart contract must perform their own due diligence.

This report should not misconstrued as a guarantee of the security properties of the smart contract.