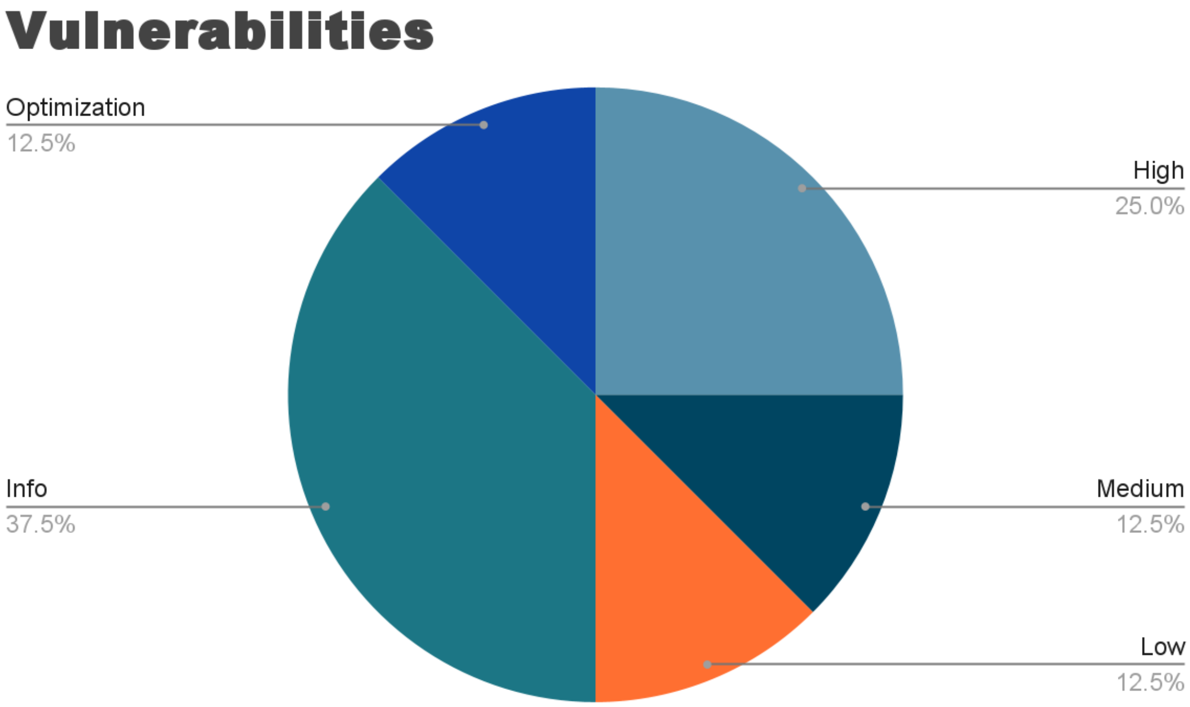
Smart-contract audit conclusion

By John Nguyen (jooohn.eth)

I was asked to review Umair Mirza's NFT contract (github) as a part of Crystalize bootcamp's assignment. The contract is tested using Slither as well as tested manually by myself. I found no critical bugs, but have discovered minor issues.



1. Findings(Slither):

ID	Severity	Subject
3.1	High	Usage of msg.value in a loop
3.2	High	Uninitialized state variables
3.3	Medium	Reentrancy Vulnerability
3.4	Low	Reentrancy Vulnerability
3.5	Info	Costly operations inside loop
3.6	Info	Conformance to Solidity naming conventions
3.7	Info	Low level calls
3.8	Optimization	Public function could be declared external

2. Introduction:

The following document provides the result of an audit. The audit's goal is a general review of smart-contract's structure, major/minor bug detections and a general recommendation.

I have audited Umair's NFT contract Github Repository. Concretely, the following file was audited:

- contracts/NFT.sol;

3. Detailed Results:

3.1 Usage of msg.value in a loop

- **Severity:** High.
- **Description**: Detected a use of `msg.value` inside a loop.
- **Recommendation**: Track `msg.value` through a local variable.
- **Slither Description:**

```
NFT.mintNft() (contracts/NFT.sol#41-49) use msg.value in a loop:
require(bool,string)(msg.value >= MINT_PRICE,Not enough funds)
(contracts/NFT.sol#42)
```

3.2 State variable shadowing

- **Severity:** High.
- **Description**: Uninitialized state variables/functions.
- **Recommendation**: Initialize `_tokenURIs` function.
- **Slither Description:**

```
NFT._tokenURIs (contracts/NFT.sol#14) is never initialized. It is used in:
        - NFT.tokenURI(uint256) (contracts/NFT.sol#58-65)
```

3.3 Reentrancy vulnerability #1

- **Severity:** Medium.
- **Description**: Detection of reentrancy bug. Potential reentrancy exploit.
- **Recommendation**: Change state variable's value before the call to external contract.
- **Slither Description:**

```
Reentrancy in NFT.mintNft() (contracts/NFT.sol#41-49):
        - s_tokenCounter = s_tokenCounter + 1 (contracts/NFT.sol#47)
```

3.4 Reentrancy vulnerability #2

- **Severity:** Low.
- **Description**: Detection of reentrancy bug. Potential reentrancy exploit.
- **Recommendation**: Change state variable's value before the call to external contract.
- **Slither Description:**

```
Reentrancy in NFT.mintNft() (contracts/NFT.sol#41-49):
        External calls:
        - _safeMint(msg.sender,s_tokenCounter) (contracts/NFT.sol#45)
(node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#401-412)
        State variables written after the call(s):
        - _setTokenURI(s_tokenCounter,TOKEN_URI) (contracts/NFT.sol#46)
                - _tokenURIs[tokenId] = _tokenURI
(node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721URIS
torage.sol#47)
```

3.5 Costly operation inside a loop

- **Severity:** Info.
- **Description**: Costly operations inside a loop might waste gas, which might lead to an out-of-gas.
- **Recommendation**: Use a local variable to hold the loop computation result. Try using the "Counters" library by Openzeppelin.
- **Slither Description:**

```
NFT.mintNft() (contracts/NFT.sol#41-49) has costly operations inside a
loop:
        - s_tokenCounter = s_tokenCounter + 1 (contracts/NFT.sol#47)
```

3.6 Conformance to Solidity naming conventions

- **Severity:** Info.
- **Description**: Variable doesn't follow Solidity's conventions.
- **Recommendation**: Follow Solidity's naming convention.
- **Slither Description:**

```
Parameter NFT.mintMultipleNfts(uint256)._count (contracts/NFT.sol#26) is
not in mixedCase:
Variable NFT.s_tokenCounter (contracts/NFT.sol#12) is not in mixedCase
```

3.7 Low level calls

- **Severity:** Info.
- **Description**: The use of low-level calls are error-prone. Low-level calls do not check for code existance or call success.
- **Recommendation**: Avoid low-level calls. Check the call success. If the call is meant for a contract, check for code existence
- **Slither Description:**

```
Low level call in NFT.withdraw() (contracts/NFT.sol#51-56):
        - (success) = (msg.sender).call{value: balance}()
(contracts/NFT.sol#54)
```

3.8 Public function could be declared external

- **Severity:** Optimization.
- **Description**: Public functions that are never called by the contract should be declared external to save gas.
- **Recommendation**: Use the external attribute for functions never called from the contract.
- **Slither Description:**

```
mintMultipleNfts(uint256) should be declared external:
        - NFT.mintMultipleNfts(uint256) (contracts/NFT.sol#26-39)
withdraw() should be declared external:
        - NFT.withdraw() (contracts/NFT.sol#51-56)
tokenURI(uint256) should be declared external:
        - NFT.tokenURI(uint256) (contracts/NFT.sol#58-65)
getTokenCounter() should be declared external:
        - NFT.getTokenCounter() (contracts/NFT.sol#67-69)
```