SOOHO

347da692245a4de84eeb9bbc62baf44238f5a5860c44c4c005245616de8ccb6

File: NFT.sol | Language:solidity | Size:721 bytes | Date:2021-04-25T07:19:41.548Z
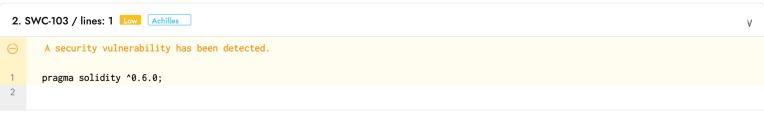
Critical	High	Medium	Low	Note
0	0	1	1	2

Issues

Severity	Issue	Analyzer	Code Lines
Medium	SWC-102	Achilles	1
Low	SWC-103	Achilles	1
Note	SWC-131	Achilles	11, 12

Code

1. SWC-102 / lines: 1   Medium   Achilles

⊖   A security vulnerability has been detected.

```
1    pragma solidity ^0.6.0;
2
```

In detail

Using an outdated compiler version can be problematic especially if there are publicly disclosed bugs and issues that affect the current compiler version.

2. SWC-103 / lines: 1   Low   Achilles

⊖   A security vulnerability has been detected.
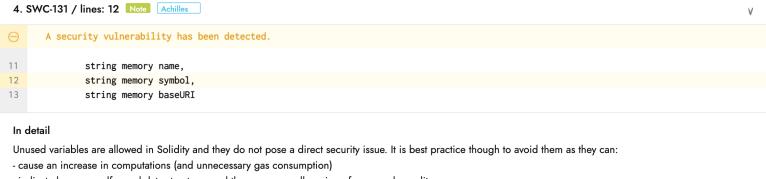
```
1    pragma solidity ^0.6.0;
2
```

In detail

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

3. SWC-131 / lines: 11   Note   Achilles

⊖   A security vulnerability has been detected.

```
10        constructor(
11            string memory name,
12            string memory symbol,
```

In detail

Unused variables are allowed in Solidity and they do not pose a direct security issue. It is best practice though to avoid them as they can:
- cause an increase in computations (and unnecessary gas consumption)
- indicate bugs or malformed data structures and they are generally a sign of poor code quality
- cause code noise and decrease readability of the code

4. SWC-131 / lines: 12 `Note` `Achilles`   ∨

⊖   A security vulnerability has been detected.

```
11          string memory name,
12          string memory symbol,
13          string memory baseURI
```

In detail

Unused variables are allowed in Solidity and they do not pose a direct security issue. It is best practice though to avoid them as they can:

- cause an increase in computations (and unnecessary gas consumption)

- indicate bugs or malformed data structures and they are generally a sign of poor code quality

- cause code noise and decrease readability of the code