Coinsult

Advanced Manual Smart Contract Audit

Project: AMCC

Website: no website

Low-Risk	Medium-Risk	High-Risk
4 low-risk code issues found	0 medium-risk code issues found	0 high-risk code issues found

Contract Address

0x5525ab1063Aa8e84f58A53F975ba5FcDD33aC494

Disclaimer

Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

Coinsult is not responsible for any financial losses. Nothing in this contract audit is financial advice, please do your own research.

The information provided in this audit is for informational purposes only and should not be considered investment advice. Coinsult does not endorse, recommend, support or suggest to invest in any project.

Coinsult can not be held responsible for when a project turns out to be a rug-pull, honeypot or scam.

Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0x0f1fdef95ae3dbb1ec6ba3165fe9c7fe99d064b9	88,893,780	88.8938%
2	Null Address: 0x000...dEaD	9,000,000	9.0000%
3	0xdb50760e6f6ffd3ec08baa6d33b6bcfa7d6ecd53	1,000,000	1.0000%
4	0x9d35bca9d7af539a27172a3e80d09d1ec6038a78	990,000	0.9900%
5	0x4884901c64a07ff6856f6ac17d711182db181454	83,939.249376681017341033	0.0839%

Source Code

Coinsult was comissioned by AMCC to perform an audit based on the following smart contract:

https://bscscan.com/address/0x5525ab1063Aa8e84f58A53F975ba5FcDD33aC494#code

Manual Code Review

In this audit report we will highlight all these issues:

🟢 Low-Risk

4 low-risk code
issues found

🟡 Medium-Risk

0 medium-risk code
issues found

🔴 High-Risk

0 high-risk code
issues found

The detailed report continues on the next page...

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Too many digits

Literals with many digits are difficult to read and review.

```
uint256 private initialAmount = 100000000;
```

Recommendation

Use: Ether suffix, Time suffix, or The scientific notation

Exploit scenario

```
contract MyContract{
    uint 1_ether = 10000000000000000000;
}
```

While `1_ether` looks like `1 ether`, it is `10 ether`. As a result, it's likely to be used incorrectly.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

No zero address validation for some functions

Detect missing zero address validation.

```
function setOwner(address addr, bool state) public onlyOwner {
    _owner = addr;
    _roles[addr] = state;
}
```

Recommendation

Check that the new address is not zero.

Exploit scenario

```
contract C {

  modifier onlyAdmin {
    if (msg.sender != owner) throw;
    _;
  }

  function updateOwner(address newOwner) onlyAdmin external {
    owner = newOwner;
  }
}
```

Bob calls `updateOwner` without specifying the `newOwner`, soBob loses ownership of the contract.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Conformance to Solidity naming conventions

Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
contract Ownable is Context {
address public _owner;
mapping(address =&gt; bool) private _roles;
```

Recommendation

Follow the Solidity naming convention.

Rule exceptions

- Allow constant variable name/symbol/decimals to be lowercase (`ERC20`).
- Allow _ at the beginning of the `mixed_case` match for private variables and unused parameters.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Redundant Statements

Detect the usage of redundant statements that have no effect.

```
function _msgData() internal view virtual returns (bytes memory) {
    this;
    // silence state mutability warning without generating bytecode - see https://github.com/ethereu
    return msg.data;
}
```

Recommendation

Remove redundant statements if they congest code but offer no value.

Exploit scenario

```
contract RedundantStatementsContract {

    constructor() public {
        uint; // Elementary Type Name
        bool; // Elementary Type Name
        RedundantStatementsContract; // Identifier
    }

    function test() public returns (uint) {
        uint; // Elementary Type Name
        assert; // Identifier
        test; // Identifier
        return 777;
    }
}
```

Each commented line references types/identifiers, but performs no action with them, so no code will be generated for such statements and they can be removed.

Owner privileges

- Owner cannot set fees higher than 25%
- Owner cannot pause trading
- Owner cannot change max transaction amount

Extra notes by the team

No notes

Contract Snapshot

```
contract AMCC is IERC20 {
address private creator = msg.sender;

uint256 public totalSupply;
string  public name;
uint8   public decimals;
string  public symbol;

uint256 private initialAmount = 100000000;
string private tokenName = "AMCC";
uint8 private decimalUnits = 18;
string private tokenSymbol = "AMCC";
```

Project Overview

Not KYC verified by Coinsult

AMCC

Audited by Coinsult.net

Date: 4 June 2022

Advanced Manual Smart Contract Audit