



Coinsult

Advanced Manual Smart Contract Audit



Project: Noodle finance

Website: <https://noodlefinance.online>

Low-Risk

3 low-risk code
issues found

Medium-Risk

0 medium-risk code
issues found

High-Risk

0 high-risk code
issues found

Contract Address

0xa24FDdb85A2cEA1281e3Af8d4F0686e39bD8A348C

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

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

Not available

Source Code

Coinsult was commissioned by Noodle finance to perform an audit based on the following smart contract:

<https://cronos.org/explorer/address/0xa24FD8b85A2cEA1281e3Af8d4F0686e39bD8A348C>

Manual Code Review

In this audit report we will highlight all these issues:

Low-Risk

3 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.

Avoid relying on `block.timestamp`

`block.timestamp` can be manipulated by miners.

```
require(unfreezeTimestamp < block.timestamp);
```

Recommendation

Do not use `block.timestamp`, `now` or `blockhash` as a source of randomness

Exploit scenario

```
contract Game {  
  
    uint reward_determining_number;  
  
    function guessing() external{  
        reward_determining_number = uint256(block.blockhash(10000)) % 10;  
    }  
}
```

Eve is a miner. Eve calls `guessing` and re-orders the block containing the transaction. As a result, Eve wins the game.

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

Too many digits

Literals with many digits are difficult to read and review.

```
uint256 private PSN = 10000;
```

Recommendation

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

Exploit scenario

```
contract MyContract{
    uint 1_ether = 1000000000000000000;
}
```

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 whitelistWallet(address wallet, bool isWhitelisted) public onlyOwner {  
    users[wallet].whitelisted = isWhitelisted;  
}
```

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, so Bob loses ownership of the contract.

Owner privileges

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

Extra notes by the team

No notes

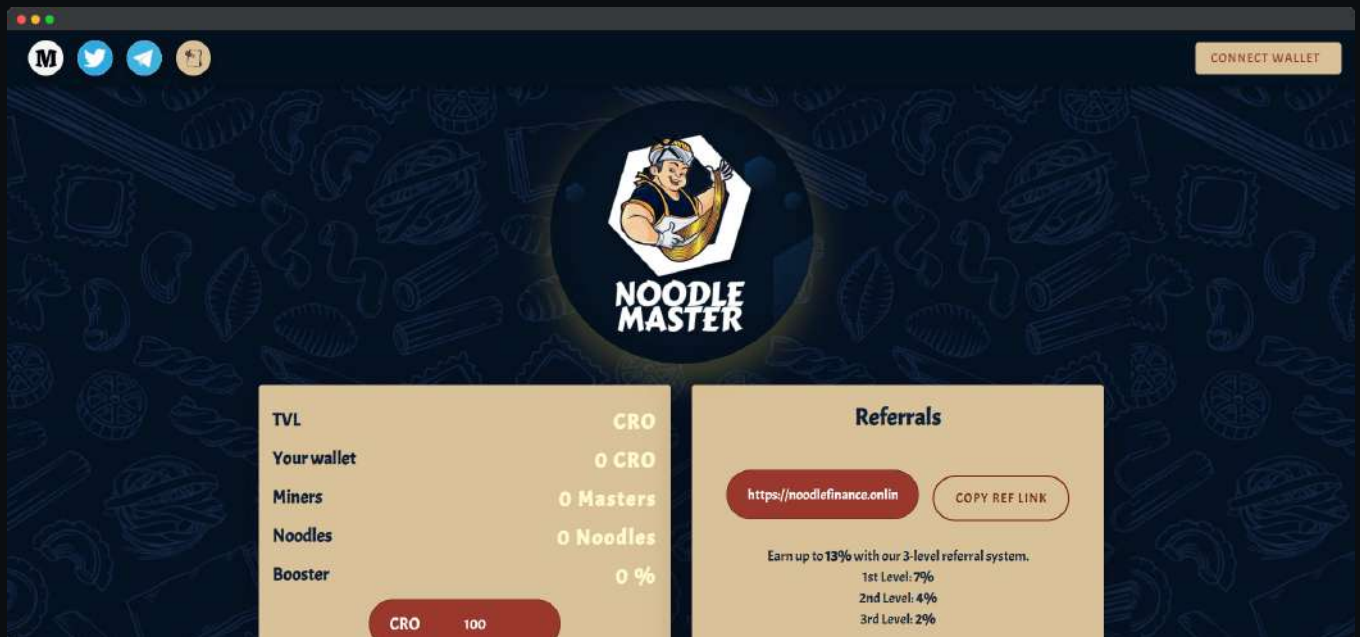
Contract Snapshot

```
contract NoodleMaster is Context, Ownable, Lottery {
    using SafeMath for uint256;

    uint256 public MAX_WHITELISTED_WALLET = 1000 * 10 ** 18; // 1000 CRO
    uint256 public CUTOFF_STEP = 48 * 60 * 60; // 48 hrs
    uint256 public UNFREEZE_TIMESPAN = 14 * 24 * 60 * 60; // 14 days
    uint256 private NODDLES_FOR_1MINER = 852637;
    uint256 public COMPOUND_STEP = 24 * 60 * 60; // 24 hrs
    uint256 public WITHDRAWAL_PENALTY_FEE = 50;
    uint256 public COMPOUND_FOR_NO_TAX_WITHDRAWAL = 5; // compounding times for no sell tax
    uint256 private PSN = 10000;
    uint256 private PSNH = 5000;
    uint256 private devFeeVal = 2;
    uint256 private partnershipFeeVal = 1;
    uint256 private marketingFeeVal = 3;
    bool private initialized = false;
    uint256 private lastActionTimestampInSec;
    address payable private devAddr;
    address payable private partnershipAddr;
    address payable private marketingAddr;
    address payable private treasuryAddr;
    mapping(address => User) private users;
    uint256 private marketNoodles;
    bool public transferActive = false;
    bool public whitelistActive = false;
    uint256 public totalWhitelistedBuys;
    uint256 public firstLevelReferralBonus = 7;
    uint256 public secondLevelReferralBonus = 4;
    uint256 public thirdLevelReferralBonus = 2;
```


Website Review

Coinsult checks the website completely manually and looks for visual, technical and textual errors. We also look at the security, speed and accessibility of the website. In short, a complete check to see if the website meets the current standard of the web development industry.



- Mobile Friendly
- Does not contain jQuery errors
- SSL Secured
- No major spelling errors

Project Overview

● Not KYC verified by Coinsult

AUDITED
BY COINSULT.NET

