Coinsult

Advanced Manual Smart Contract Audit

Project: Vulcano Vesting

Website: https://www.vulcano.gg

This audit is for the vesting contract, full-audit report of the main token is located on the Coinsult github

🟢 Low-risk	🟡 Medium-risk	🔴 High-risk
5 low-risk code issues found	0 medium-risk code issues found	0 high-risk code issues found

Contract address

Not deployed yet

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 deployed yet

Source code

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

https://github.com/NWCrypto/vulcano/blob/vesting/packages/hardhat/contracts/vesting/TokenVestingContract.sol (PRIVATE)

Note: This project uses openzeppelin imports. While we do check the full contract for vulnerabilities at the time of the audit, we can not ensure the correctness of these imported modules.

Manual Code Review

🟢 Low-risk

5 low-risk code issues found.
Could be fixed, will not bring problems.

- Avoid relying on block.timestamp. block.timestamp can be manipulated by miners.

```
function currentTime() internal view virtual returns (uint256) {
        return block.timestamp;
    }
```

- Contract contains Reentrancy vulnerabilities

  Additional information: This combination may be justified by some complex mechanics (e.g. rebase, reflections, buyback). More information: Slither

```
contract TokenVestingContract is Pausable, AccessControl,
ReentrancyGuard {
    using SafeMath for uint256;
    using SafeERC20 for IERC20;

    // Role-related constants
    bytes32 public constant PAUSER_ROLE = keccak256("PAUSER_ROLE");
    bytes32 public constant GRANTOR_ROLE = keccak256("GRANTOR_ROLE");

    // Date-related constants
    uint256 private constant TEN_YEARS_DAYS = 10 * 365;
    uint256 private constant SECONDS_PER_DAY = 24 * 60 * 60;

    // Structs
    struct VestingSchedule {
        bool isValid;              /* true if an entry exists and is
valid. */
        uint256 cliffDuration;     /* Duration of the cliff, with
respect to the grant start day, in days. */
        uint256 duration;          /* Duration of the vesting
schedule, with respect to the grant start time, in days. */
```

- Contract can be paused and unpaused by `'PAUSER_ROLE'`

```
function pause() public onlyRole(PAUSER_ROLE) {
    _pause();
}

function unpause() public onlyRole(PAUSER_ROLE) {
    _unpause();
}
```

- Combine code to create a shorter contract

  Additional information: 'elapsedTime' is never used, only once to create a new variable. You could create only elapsedDays.

```
uint256 elapsedTime = currentTime().sub(tokenGrant.startTime);
uint256 elapsedDays = elapsedTime.div(SECONDS_PER_DAY);
```

- Hard coded date values and leap years

  While this can be intentional, leap years do not have 365 days per year. This might create unintentional errors.

```
// Date-related constants
uint256 private constant TEN_YEARS_DAYS = 10 * 365;
```

  🟢 Changed by the team after the audit

🟡 Medium-risk

0 medium-risk code issues found.
Should be fixed, could bring problems.

🔴 High-risk

0 high-risk code issues found
Must be fixed, and will bring problems.

Extra notes by the team

Note: This project uses openzeppelin imports. While we do check the full contract for vulnerabilities at the time of the audit, we can not ensure the correctness of these imported modules.

- 🟢 This contract is a token vesting contract

- 🟡 Dev notes should be removed before launch
  - 🟢 Changed by the team after the audit

Contract Snapshot

```
contract TokenVestingContract is Pausable, AccessControl,
ReentrancyGuard {
    using SafeMath for uint256;
    using SafeERC20 for IERC20;

    // Role-related constants
    bytes32 public constant PAUSER_ROLE = keccak256("PAUSER_ROLE");
    bytes32 public constant GRANTOR_ROLE = keccak256("GRANTOR_ROLE");

    // Date-related constants
    uint256 private constant TEN_YEARS_DAYS = 10 * 365;
    uint256 private constant SECONDS_PER_DAY = 24 * 60 * 60;

    // Structs
    struct VestingSchedule {
        bool isValid; /* true if an entry exists and is valid. */
        uint256 cliffDuration; /* Duration of the cliff, with respect
to the grant start day, in days. */
        uint256 duration; /* Duration of the vesting schedule, with
respect to the grant start time, in days. */
    }

    struct TokenGrant {
        bool isValid; /* true if an entry exists and is valid. */
        bool wasRevoked; /* true if this token grant was revoked. */
        address beneficiary; /* The beneficiary of this token grant */
        bytes32 vestingScheduleId; /* Id of the vesting schedule
associated with this token grant */
        uint256 startTime; /* Start time of the token grant. */
        uint256 daysClaimed; /* Days already claimed. */
        uint256 tokensAmount; /* Total number of tokens that vest. */
        uint256 tokensClaimed; /* Total number of tokens claimed. */
    }
```

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
- 🟢 Contains no jQuery errors
- 🟢 SSL Secured
- 🟢 No major spelling errors

Note: The website is graphically dependent on the visitor's graphics card; The website might lag if the visitor has an outdated graphics card.

Loading speed: 88%

Rug-pull Review

Based on the available information analyzed by us, we come to the following conclusions:

🟢 Locked Liquidity (no liquidity yet)

🟢 Large unlocked wallets

- Note: Tokens not distributed yet

🟢 Doxxed Team

Honeypot Review

Based on the available information analyzed by us, we come to the following conclusions:

🟢 Ability to sell

🔴 Owner is able to pause the contract

🟢 Router not hard coded in the contract

Note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by the project owner.