Coinsult

Advanced Manual Smart Contract Audit

Project: Python

Website: http://www.pyhons.com

🟢 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

0x7E284A0D6fEc2ac002c9BB0B7eCc53a4a4f1Fde0

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	0x95782119b0dac5fee96187376459f780df586811	770,000,000	77.0000%
2	0x08df3400805627e42d4ec33cfaf9d040e28cb0f8	200,000,000	20.0000%
3	0x62da7160099b54b9fef530bab5c8c11a6795ba62	30,000,000	3.0000%

Source Code

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

https://bscscan.com/address/0x7e284a0d6fec2ac002c9bb0b7ecc53a4a4f1fde0#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.

Contract contains Reentrancy vulnerabilities

Additional information: This combination increases risk of malicious intent. While it may be justified by some complex mechanics (e.g. rebase, reflections, buyback).

More information: Slither

```
function _transfer(
    address from,
    address to,
    uint256 amount
) private {
    require(from != address(0), "ERC20: transfer from the zero address");
    require(amount &gt; 0, "Transfer amount must be greater than zero");

    uint256 senderBalance = _balances[from];
    require(senderBalance &gt;= amount, "ERC20: transfer amount exceeds balance");

    // transfer amount
    // is the token balance of this contract address over the min number of
    // tokens that we need to initiate a swap + liquidity lock?
    // also, don't get caught in a circular liquidity event.
    // also, don't swap &amp; liquify if sender is uniswap pair.
    uint256 contractTokenBalance = balanceOf(address(this));
    bool overMinTokenBalance = contractTokenBalance &gt;= _minTokenBalance;

    if (
        !currentlySwapping &amp;&amp;
        overMinTokenBalance &amp;&amp;
```

Recommendation

Apply the check-effects-interactions pattern.

Exploit scenario

```
function withdrawBalance(){
    // send userBalance[msg.sender] Ether to msg.sender
    // if mgs.sender is a contract, it will call its fallback function
    if( ! (msg.sender.call.value(userBalance[msg.sender])() ) ){
        throw;
    }
    userBalance[msg.sender] = 0;
}
```

Bob uses the re-entrancy bug to call withdrawBalance two times, and withdraw more than its initial deposit to the contract.

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

Too many digits

Literals with many digits are difficult to read and review.

```
uint256 private _totalSupply = 10 * 100000000 * 10**9;
```

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.

Conformance to Solidity naming conventions

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

```
uint256 public _MarketFee = 3;
```

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/et
    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
- 🟡 Owner can exclude from fees

Extra notes by the team

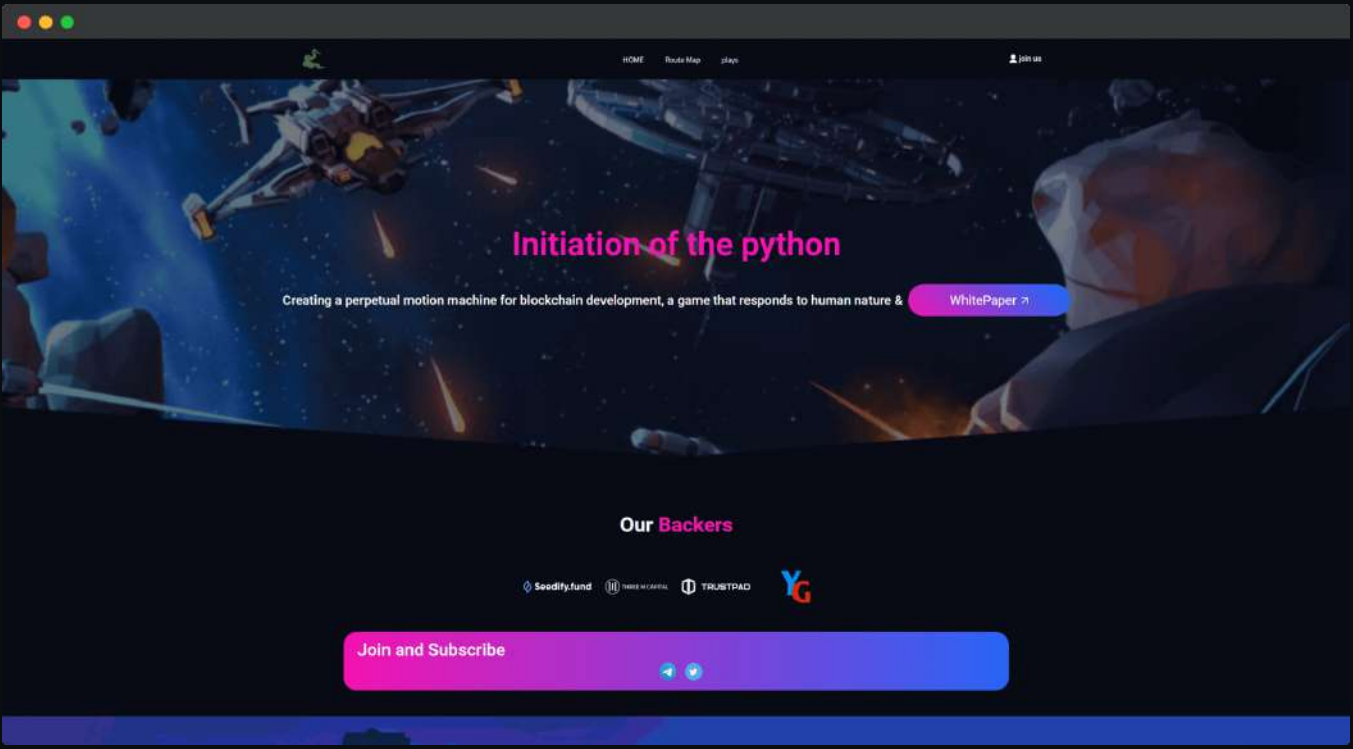
No notes

Contract Snapshot

```
contract PYTHON is Context, IERC20, Ownable {
using SafeMath for uint256;
using Address for address;

mapping (address =&gt; uint256) private _balances;
mapping (address =&gt; mapping (address =&gt; uint256)) private _allowances;
mapping (address =&gt; bool) private _isExcludedFromFee;
mapping (address =&gt; bool) private _isExcludedFromPair;

uint256 private _totalSupply = 10 * 100000000 * 10**9;

string private _name = "PYTHON";
string private _symbol = "PYH";
uint8 private _decimals = 9;

uint256 public _burnFee = 2;
uint256 public _MarketFee = 3;
```

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.



Creating a perpetual motion machine for blockchain development, a game that responds to human nature &

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

Note:
Website is not forced to SSL

Project Overview

● Not KYC verified by Coinsult