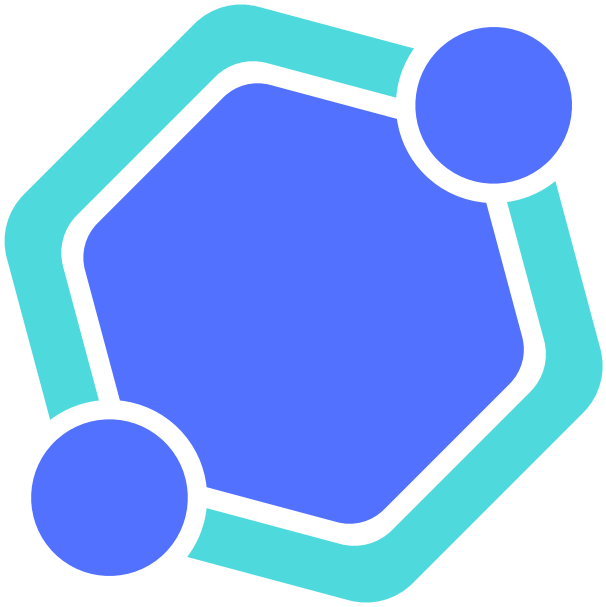
AUDIT REPORT

December 2023



Audit conducted by

RICARDO PONTES

Summary

Auditing Firm	Crypto Hub
Architecture	Crypto Hub Auditing Standard
Smart Contract Audit Approved By	Ricardo | Blockchain Dev at Crypto Hub
Platform	Solidity
Mandatory Audit Check	Static, Software & Manual Analysis
Consultation Request Date	December 2, 2023
Report Date	December 3, 2023

Audit Summary

Crypto Hub team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

- ★ ALTITUDE smart contract source code has **LOW RISK SEVERITY**.
- ★ ALTITUDE has **PASSED** the smart contract audit.

For the detailed understanding of risk severity, source code vulnerability, and functional test, kindly refer to the audit.

✅ Verify the authenticity of this report on Crypto Hub Website: https://www.cryptohub.agency/

Table Of Contents

Project Overview

Crypto Hub was consulted by ALTITUDE to conduct the smart contract security audit of their solidity source code.

Project	ALTITUDE
Blockchain	Base Mainnet
Language	Solidity
Contracts	0x72b2fd2a6964aaa6233336751996a28a9568a40d
Website:	https://www.altitudetoken.com/

Public logo:

Solidity Source Code On Blockchain (Verified Contract Source Code)

https://basescan.org/token/0x90678c02823b21772fa7e91b27ee70490257567b#code

Contract Name: ALTITUDE

Compiler Version: v0.8.18

Optimization Enabled: yes

SHA-1 Hash

Solidity source code is audited at hash

#27ef8d3093ec8ad14f0e8ed66625e61cfa5167b3

Audit Scope & Methodology

The scope of this report is to audit the above smart contract source code and Crypto Hub has scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Smart Contract Vulnerabilities

- ☐ Re-entrancy
- ☐ Unhandled Exceptions
- ☐ Transaction Order Dependency
- ☐ Integer Overflow
- ☐ Unrestricted Action
- ☐ Incorrect Inheritance Order
- ☐ Typographical Errors
- ☐ Requirement Violation

Source Code Review

- ☐ Ownership Takeover
- ☐ Gas Limit and Loops
- ☐ Deployment Consistency
- ☐ Repository Consistency
- ☐ Data Consistency
- ☐ Token Supply Manipulation

Functional Assessment

- ☐ Access Control and Authorization
- ☐ Operations Trail and Event Generation
- ☐ Assets Manipulation
- ☐ Liquidity Access

Crypto Hub Audit Standard

The aim of Crypto Hub standard is to analyze the smart contract and identify the vulnerabilities and the hacks in the smart contract. Mentioned are the steps used by ECHELON-1 to assess the smart contract:

1. Solidity smart contract source code reviewal:
   - Review of the specifications, sources, and instructions provided to Crypto Hub to make sure we understand the size, scope, and functionality of the smart contract.
   - Manual review of code, which is the process of reading source code line-by-line to identify potential vulnerabilities.

2. Static, Manual, and Software analysis:
   - Test coverage analysis, which is the process of determining whether the test cases are covering the code and how much code is exercised when we run those test cases.
   - Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.

3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts

Automated 3P frameworks used to assess the smart contract vulnerabilities

- Slither
- Consensys MythX
- Consensus Surya
- Open Zeppelin Code Analyzer
- Solidity Code Complier

Crypto Hub's Risk Classification

Smart contracts are generally designed to manipulate and hold funds denominated in ETH/BNB. This makes them very tempting attack targets, as a successful attack may allow the attacker to directly steal funds from the contract. Below are the typical risk levels of a smart contract: Vulnerable:

A contract is vulnerable if it has been flagged by a static analysis tool as such. As we will see later, this means that some contracts may be vulnerable because of a false-positive.

Exploitable: A contract is exploitable if it is vulnerable and the vulnerability could be exploited by an external attacker. For example, if the "vulnerability" flagged by a tool is in a function which requires to own the contract, it would be vulnerable but not exploitable.

Exploited: A contract is exploited if it received a transaction on the main network which triggered one of its vulnerabilities. Therefore, a contract can be vulnerable or even exploitable without having been exploited.

Risk severity	Meaning
! Critical	This level of vulnerability could be exploited easily, and can lead to asset loss, data loss, asset manipulation, or data manipulation. They should be fixed right away.
! High	This level vulnerabilities are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to critical risk severity
! Medium	This level of vulnerabilities should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution.
! Low	This level of vulnerability can be ignored. They are code style violations, and informational statements in the code. They may not affect the smart contract execution

Smart Contract Risk Assessment

Contract Snapshot

```
contract ALTITUDE is IERC20, Ownable {
    using SafeERC20 for IERC20;
    uint8 private _decimals;
    mapping(address => uint256) private _rOwned;
    mapping(address => uint256) private _tOwned;
    mapping(address => mapping(address => uint256)) private _allowances;
    mapping(address => bool) private _isExcluded;
    address[] private _excluded;
    uint256 private constant MAX = ~uint256(0);
    uint256 private _tTotal;
    uint256 private _rTotal;
    uint256 private _tFeeTotal;
    string private _name;
    string private _symbol;

    uint256 private _rewardFee;
    uint256 private _previousRewardFee;

    uint256 private _liquidityFee;
    uint256 private _previousLiquidityFee;

    uint256 private _marketingFee;
    uint256 private _previousMarketingFee;

    uint256 private _burnFee;
    uint256 private _previousBurnFee;

    bool private inSwapAndLiquify;
    uint16 public sellRewardFee;
    uint16 public buyRewardFee;
    uint16 public sellLiquidityFee;
    uint16 public buyLiquidityFee;

    uint16 public sellMarketingFee;
    uint16 public buyMarketingFee;

    uint16 public sellBurnFee;
    uint16 public buyBurnFee;

    address public marketingWallet;
    bool public isMarketingFeeETH;

    uint256 public minAmountToTakeFee;
    uint256 public maxWallet;
    uint256 public maxTransactionAmount;


    IUniswapV2Router02 public mainRouter;
    address public mainPair;

    mapping(address => bool) public isExcludedFromFee;
    mapping(address => bool) public isExcludedFromMaxTransactionAmount;
    mapping(address => bool) public automatedMarketMakerPairs;

    uint256 private _liquidityFeeTokens;
    uint256 private _marketingFeeTokens;
```

Static / Quick Analysis

Honeypot Risk

Buy Tax: unknown Sell Tax: unknown

This does not appear to be a honeypot.

We are not aware of any malicious code.

No codes found to suspend trading.

If a suspendable code is included, the token maybe neither be bought nor sold (honeypot risk).

The token can be bought

Generally, these unbuyable tokens would be found in Reward Tokens. Such Tokens are issued as rewards for some on-chain applications and cannot be bought directly by users.

No trading cooldown function

The token contract has no trading cooldown function. If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying.

Anti_whale(Limited number of transactions)

The number of token transactions is limited. The number of scam token transactions may be limited (honeypot risk).

Anti whale is modifiable

The maximum token trading amount or maximum position can be modified, which may lead to suspension of trading. (honeypot risk).

Tax can be modified

The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens will not be able to be traded (honeypot risk).

No blacklist

The blacklist function is not included. If there is a blacklist, some addresses may not be able to trade normally (honeypot risk).

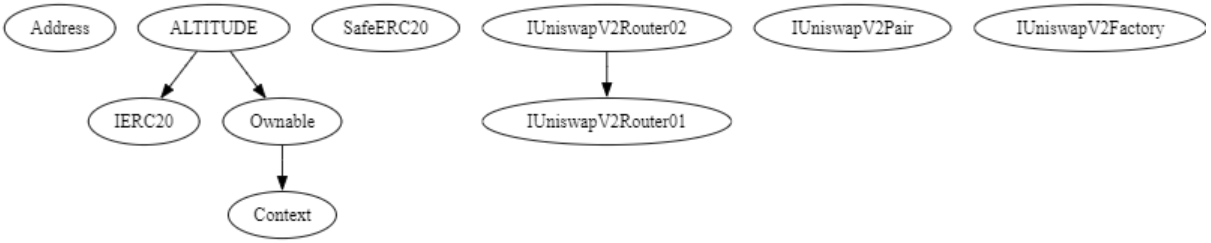
Whitelist function

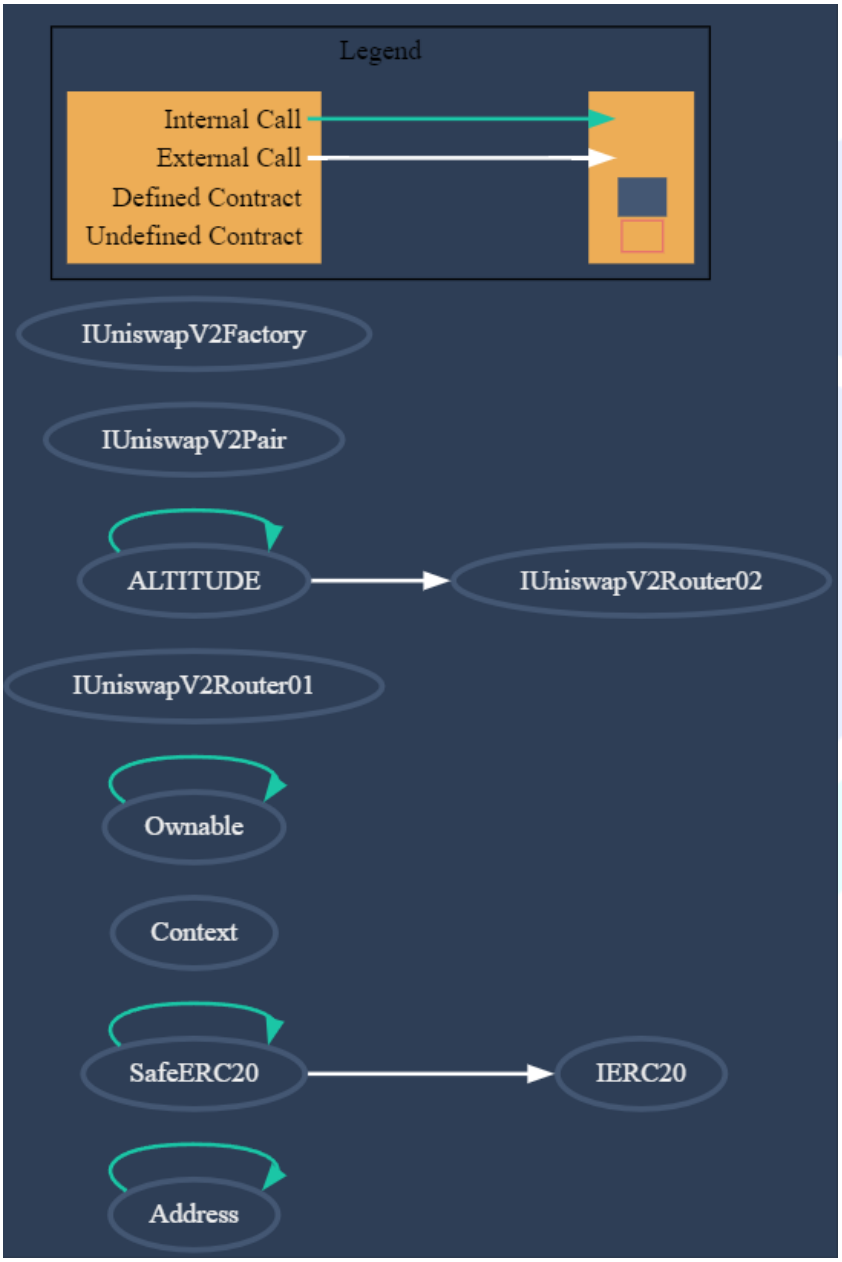
The whitelist function is included. Some addresses may not be able to trade normally (honeypot risk).

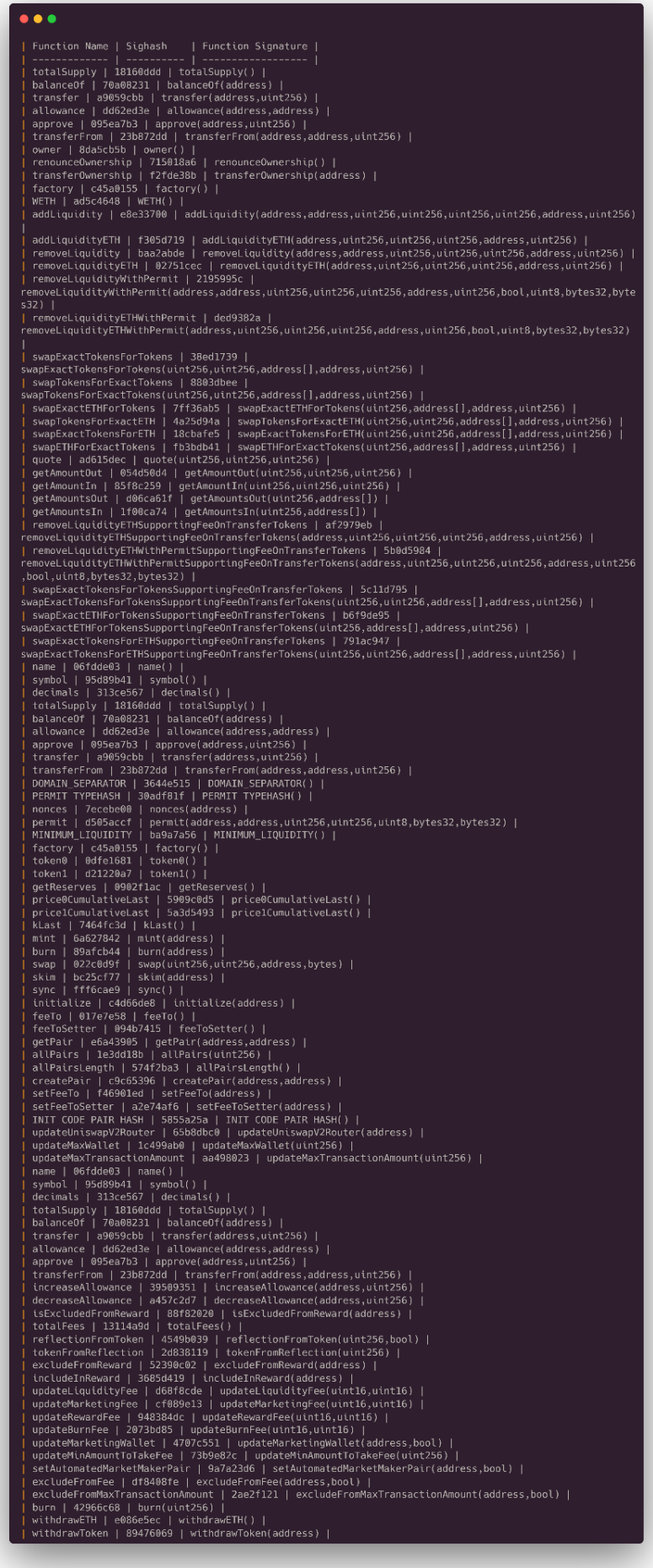
No tax changes found for personal addresses

No tax changes were found for every assigned address. If it exists, the contract owner may set a very outrageous tax rate for assigned address to block it from trading.

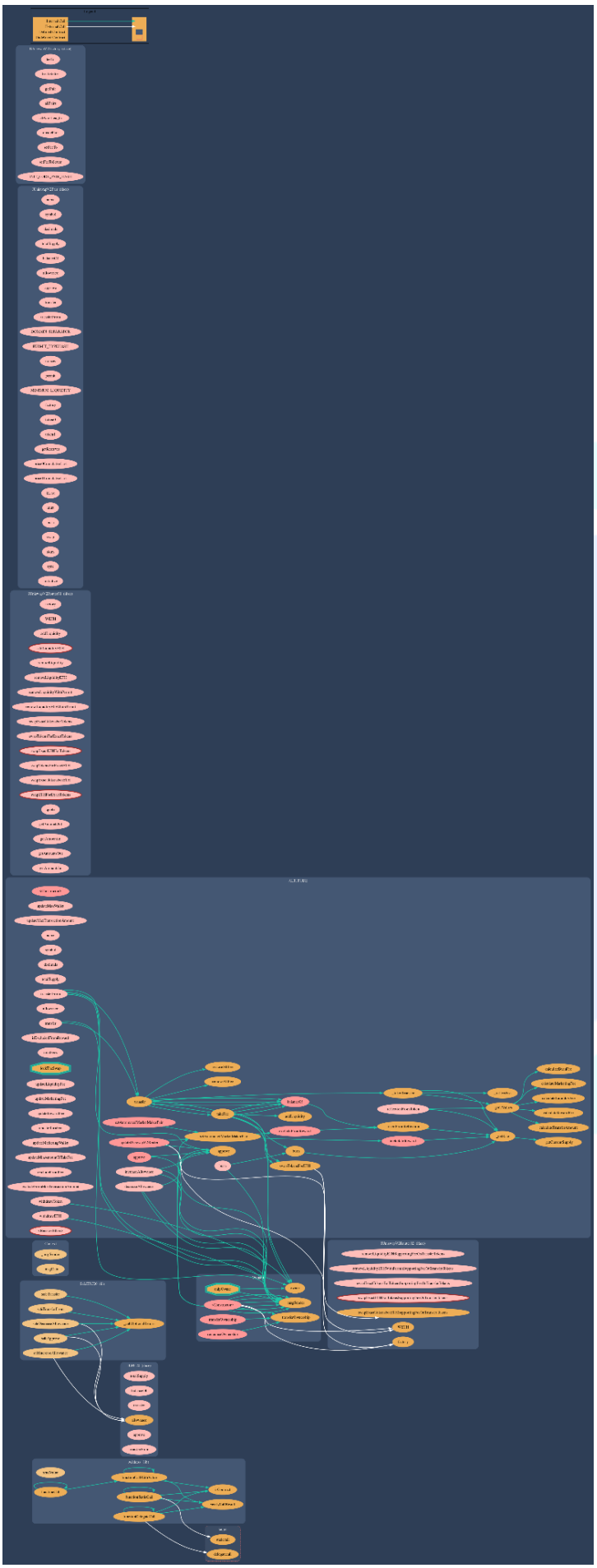
Software Analysis







Function Name	Sighash	Function Signature
totalSupply	18160ddd	totalSupply()
balanceOf	70a08231	balanceOf(address)
transfer	a9059cbb	transfer(address,uint256)
allowance	dd62ed3e	allowance(address,address)
approve	095ea7b3	approve(address,uint256)
transferFrom	23b872dd	transferFrom(address,address,uint256)
owner	8da5cb5b	owner()
renounceOwnership	715018a6	renounceOwnership()
transferOwnership	f2fde38b	transferOwnership(address)
factory	c45a0155	factory()
WETH	ad5c4648	WETH()
addLiquidity	e8e33700	addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256)
addLiquidityETH	f305d719	addLiquidityETH(address,uint256,uint256,uint256,address,uint256)
removeLiquidity	baa2abde	removeLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256)
removeLiquidityETH	02751cec	removeLiquidityETH(address,uint256,uint256,uint256,address,uint256)
removeLiquidityWithPermit	2195995c	removeLiquidityWithPermit(address,address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes32,bytes32)
removeLiquidityETHWithPermit	ded9382a	removeLiquidityETHWithPermit(address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes32,bytes32)
swapExactTokensForTokens	38ed1739	swapExactTokensForTokens(uint256,uint256,address[],address,uint256)
swapTokensForExactTokens	8803dbee	swapTokensForExactTokens(uint256,uint256,address[],address,uint256)
swapExactETHForTokens	7ff36ab5	swapExactETHForTokens(uint256,address[],address,uint256)
swapTokensForExactETH	4a25d94a	swapTokensForExactETH(uint256,uint256,address[],address,uint256)
swapExactTokensForETH	18cbafe5	swapExactTokensForETH(uint256,uint256,address[],address,uint256)
swapETHForExactTokens	fb3bdb41	swapETHForExactTokens(uint256,address[],address,uint256)
quote	ad615dec	quote(uint256,uint256,uint256)
getAmountOut	054d50d4	getAmountOut(uint256,uint256,uint256)
getAmountIn	85f8c259	getAmountIn(uint256,uint256,uint256)
getAmountsOut	d06ca61f	getAmountsOut(uint256,address[])
getAmountsIn	1f00ca74	getAmountsIn(uint256,address[])
removeLiquidityETHSupportingFeeOnTransferTokens	af2979eb	removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,uint256,uint256,address,uint256)
removeLquidityETHWithPermitSupportingFeeOnTransferTokens	5b0d5984	removeLquidityETHWithPermitSupportingFeeOnTransferTokens(address,uint256,uint256,uint256,address,uint256,bool,uint8,bytes32,bytes32)
swapExactTokensForTokensSupportingFeeOnTransferTokens	5c11d795	swapExactTokensForTokensSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)
swapExactETHForTokensSupportingFeeOnTransferTokens	b6f9de95	swapExactETHForTokensSupportingFeeOnTransferTokens(uint256,address[],address,uint256)
swapExactTokensForETHSupportingFeeOnTransferTokens	791ac947	swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)
name	06fdde03	name()
symbol	95d89b41	symbol()
decimals	313ce567	decimals()
totalSupply	18160ddd	totalSupply()
balanceOf	70a08231	balanceOf(address)
allowance	dd62ed3e	allowance(address,address)
approve	095ea7b3	approve(address,uint256)
transfer	a9059cbb	transfer(address,uint256)
transferFrom	23b872dd	transferFrom(address,address,uint256)
DOMAIN_SEPARATOR	3644e515	DOMAIN_SEPARATOR()
PERMIT_TYPEHASH	30adf81f	PERMIT_TYPEHASH()
nonces	7ecebe00	nonces(address)
permit	d505accf	permit(address,address,uint256,uint256,uint8,bytes32,bytes32)
MINIMUM_LIQUIDITY	ba9a7a56	MINIMUM_LIQUIDITY()
factory	c45a0155	factory()
token0	0dfe1681	token0()
token1	d21220a7	token1()
getReserves	0902f1ac	getReserves()
price0CumulativeLast	5909c0d5	price0CumulativeLast()
price1CumulativeLast	5a3d5493	price1CumulativeLast()
kLast	7464fc3d	kLast()
mint	6a627842	mint(address)
burn	89afcb44	burn(address)
swap	022c0d9f	swap(uint256,uint256,address,bytes)
skim	bc25cf77	skim(address)
sync	fff6cae9	sync()
initialize	c4d66de8	initialize(address)
feeTo	017e7e58	feeTo()
feeToSetter	094b7415	feeToSetter()
getPair	e6a43905	getPair(address,address)
allPairs	1e3dd18b	allPairs(uint256)
allPairsLength	574f2ba3	allPairsLength()
createPair	c9c65396	createPair(address,address)
setFeeTo	f46901ed	setFeeTo(address)
setFeeToSetter	a2e74af6	setFeeToSetter(address)
INIT_CODE_PAIR_HASH	5855a25a	INIT_CODE_PAIR_HASH()
updateUniswapV2Router	65b8dbc0	updateUniswapV2Router(address)
updateMaxWallet	1c499ab0	updateMaxWallet(uint256)
updateMaxTransactionAmount	aa498023	updateMaxTransactionAmount(uint256)
name	06fdde03	name()
symbol	95d89b41	symbol()
decimals	313ce567	decimals()
totalSupply	18160ddd	totalSupply()
balanceOf	70a08231	balanceOf(address)
transfer	a9059cbb	transfer(address,uint256)
allowance	dd62ed3e	allowance(address,address)
approve	095ea7b3	approve(address,uint256)
transferFrom	23b872dd	transferFrom(address,address,uint256)
increaseAllowance	39509351	increaseAllowance(address,uint256)
decreaseAllowance	a457c2d7	decreaseAllowance(address,uint256)
isExcludedFromReward	88f82020	isExcludedFromReward(address)
totalFees	13114a9d	totalFees()
reflectionFromToken	4549b039	reflectionFromToken(uint256,bool)
tokenFromReflection	2d838119	tokenFromReflection(uint256)
excludeFromReward	52390c02	excludeFromReward(address)
includeInReward	3685d419	includeInReward(address)
updateLiquidityFee	d68f8cde	updateLiquidityFee(uint16,uint16)
updateMarketingFee	cf089e13	updateMarketingFee(uint16,uint16)
updateRewardFee	948384dc	updateRewardFee(uint16,uint16)
updateBurnFee	2073bd85	updateBurnFee(uint16,uint16)
updateMarketingWallet	4707c551	updateMarketingWallet(address,bool)
updateMinAmountToTakeFee	73b9e82c	updateMinAmountToTakeFee(uint256)
setAutomatedMarketMakerPair	9a7a23d6	setAutomatedMarketMakerPair(address,bool)
excludeFromFee	df8408fe	excludeFromFee(address,bool)
excludeFromMaxTransactionAmount	2ae2f121	excludeFromMaxTransactionAmount(address,bool)
burn	42966c68	burn(uint256)
withdrawETH	e086e5ec	withdrawETH()
withdrawToken	89476069	withdrawToken(address)



SWC Attacks

The following table contains an overview of the SWC registry. Each row consists of an SWC identifier (ID), weakness title, CWE parent and list of related code samples.

The auditor used a MythX tool, A static analyzer that parses the Soldity AST, a symbolic analyzer that detects possible vulnerable states, and a greybox fuzzer that detects vulnerable execution paths.

ID	Description	Status
SWC - 100	Function Default Visibility	✔️ **Passed**
SWC - 101	Integer Overflow and Underflow	✔️ **Passed**
SWC - 102	Outdated Compiler Version	✔️ **Passed**
SWC - 103	Floating Pragma	✔️ **Passed**
SWC - 104	Unchecked Call Return Value	✔️ **Passed**
SWC - 105	Unprotected Ether Withdrawal	✔️ **Passed**
SWC - 106	Unprotected SELFDESTRUCT Instruction	✔️ **Passed**
SWC - 107	Reentrancy Passed	✔️ **Passed**
SWC - 108	State Variable Default Visibility	✔️ **Passed**
SWC - 109	Uninitialized Storage Pointer	✔️ **Passed**
SWC - 110	Assert Violation Passed	✔️ **Passed**
SWC - 111	Use of Deprecated Solidity Functions	✔️ **Passed**
SWC - 112	Delegatecall to Untrusted Callee	✔️ **Passed**
SWC - 113	DoS with Failed Call	✔️ **Passed**
SWC - 114	Transaction Order Dependence	✔️ **Passed**
SWC - 115	Authorization through tx.origin	✔️ **Passed**
SWC - 116	Block values as a proxy for time	✔️ **Passed**
SWC - 117	Signature Malleability	✔️ **Passed**

ID	Description	Status
SWC - 118	Incorrect Constructor Name	✔️ **Passed**

SWC - 119	Shadowing State Variables	✔️ **Passed**
SWC - 120	Weak Sources of Randomness from Chain Attributes	✔️ **Passed**
SWC - 121	Missing Protection against Signature Replay Attacks	✔️ **Passed**
SWC - 122	Lack of Proper Signature Verification	✔️ **Passed**
SWC - 123	Requirement Violation Passed	✔️ **Passed**
SWC - 124	Write to Arbitrary Storage Location	✔️ **Passed**
SWC - 125	Incorrect Inheritance Order Passed	✔️ **Passed**
SWC - 126	Insufficient Gas Griefing	✔️ **Passed**
SWC - 127	Arbitrary Jump with Function Type Variable	✔️ **Passed**
SWC - 128	DoS With Block Gas Limit	✔️ **Passed**
SWC - 129	Typographical Error	✔️ **Passed**
SWC - 130	Right-To-Left-Override control character (U+202E)	✔️ **Passed**
SWC - 131	Presence of unused variables	✔️ **Passed**
SWC - 132	Unexpected Ether balance	✔️ **Passed**
SWC - 133	Hash Collisions With Multiple Variable Arguments	✔️ **Passed**

SWC - 134	Message call with hardcoded gas amount	✔️ **Passed**
SWC - 135	Code With No Effects	✔️ **Passed**
SWC - 136	Unencrypted Private Data On-Chain	✔️ **Passed**

Manual Analysis

Risk Status

Risk severity	Meaning
! Critical	None critical severity issues identified
! High	None high severity issues identified
! Medium	None medium severity issues identified
! Low	None low severity issues identified
Verified	29 functions and instances verified and checked
Safety Score	93 out of 100

Report Summary

Crypto Hub team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks.

ALTITUDE smart contract source code has **LOW RISK SEVERITY.**
ALTITUDE has **PASSED** the smart contract audit.

Note for stakeholders:

Be aware that active smart contract owner privileges constitute an elevated impact on smart contract's safety and security.
Make sure that the project team's KYC/identity is verified by an independent firm, e.g., Crypto Hub.
Always check if the contract's liquidity is locked. A longer liquidity lock plays an important role in the project's longevity. It is recommended to have multiple liquidity providers.
Examine the unlocked token supply in the owner, developer, or team's private wallets. Understand the project's tokenomics, and make sure the tokens outside of the LP Pair are vested or locked for a longer period of time.
Ensure that the project's official website is hosted on a trusted platform, and is using an active SSL certificate. The website's domain should be registered for a longer period of time.

Audit & KYC Certificates

We hereby certificate ALTITUDE smart contract as an audited project under the Crypto Hub enterprise umbrella. And to represent it as such we issued the following certificate:

Legal Advisory

Important Disclaimer

Crypto Hub provides contract auditing and project verification services for blockchain projects. The purpose of the audit is to analyze the on-chain smart contract source code, and to provide a basic overview of the project. This report should not be transmitted, disclosed, referred to, or relied upon by any person for any purposes without Crypto Hub prior written consent.

Crypto Hub provides the easy-to-understand assessment of the project, and the smart contract (otherwise known as the source code). The audit makes no statements or warranties on the security of the code. It also cannot be considered as an adequate assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. While we have used all the data at our disposal to provide the transparent analysis, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. Be aware that smart contracts deployed on a blockchain aren't resistant from external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security. Therefore, Crypto Hub does not guarantee the explicit security of the audited smart contract.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

This report should not be considered as an endorsement or disapproval of any project or team. The information provided in this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. Do conduct your own due diligence and consult your financial advisor before making any investment decisions.

About Crypto Hub

Crypto Hub provides intelligent blockchain solutions. Crypto Hub is developing an ecosystem that is seamless and responsive. Some of our services: Blockchain Security, Token Launchpad, NFT Marketplace, etc. Crypto Hub's mission is to interconnect multiple services like Blockchain Security, DeFi, Gaming, and Marketplace under one ecosystem that is seamless, multi-chain compatible, scalable, secure, fast, responsive, and easy-to-use.

Crypto Hub is built by a decentralized team of UI experts, contributors, engineers, and enthusiasts from all over the world. Our team currently consists of 3+ core team members, and 6+ casual contributors. Crypto Hub provides manual, static, and automatic smart contract analysis, to ensure that the project is checked against known attacks and potential vulnerabilities.