



BLOCK SOLUTIONS

Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts



Request Date: 2025-01-14

Completion Date: 2025-01-15

Language: Solidity



Contents

| | |
|---|----|
| Commission | 3 |
| Sale, Presale & Fair Launch Factory Contracts Properties..... | 4 |
| Checklist..... | 5 |
| Testing Summary | 7 |
| Quick Stats: | 8 |
| Executive Summary | 9 |
| Code Quality | 9 |
| Documentation | 9 |
| Use of Dependencies..... | 9 |
| Audit Findings | 10 |
| Critical | 10 |
| High | 10 |
| Medium..... | 10 |
| Low | 10 |
| Conclusion | 11 |
| Our Methodology..... | 11 |



Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts

Commission

| | |
|-------------------------------------|---|
| Audited Project | Sale, Presale & Fair Launch Factory Smart Contracts |
| Sale Factory Contract | 0xFEFC1b487bE52A9ED92e43C224764a647C8Eb38c |
| Presale Factory Contract | 0x5c75f975A42D2750FF6f0ecD9441b518Fdb06164 |
| Fair Launch Factory Contract | 0x09B4717bA7A433C4E2b8fa9748aA3080b0E9B5F5 |
| Contracts Owner | 0x26C7DaE7378929d2A94F74cCE3139876D9c21CF4 |
| Smart Contracts Creator | 0x3332F42de35e8614416E9852983751C9cA537847 |
| Blockchain | BSC Smart Chain Mainnet |

Block Solutions was commissioned by SALE, PRESALE & FAIR LAUNCH FACTORY Smart Contracts owners to perform an audit of their main smart contracts. The purpose of the audit was to achieve the following:

- Ensure that the smart contracts functions as intended.
- Identify potential security issues with the smart contracts.

The information in this report should be used to understand the risk exposure of the smart contracts, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.



**Sale, Presale & Fair
Launch Factory Contracts
Properties**

| | |
|-------------------------------------|--|
| Token Fee | 0.25 ETH |
| Fee Receiver | 0x15C1237FBd4a93d09D9318faF055F085aBf29ca1 |
| Lock Address | 0xf6692bCe99e5c462A1d948d8d96aa32E0c5C79a4 |
| Emergency Fee Receiver | 0x72269c514C9c6B66C451d7Edd15bE53eebcAAf7A |
| Creation Fee Receiver | 0x1FEB807C60331E6196C8BBD177A3902A35426610 |
| Creation Fee | 0.5 BNB |
| Admin | 0x761034F9D3B8B193088CF8a3FA6c96E8F3b650ce |
| Base Factory | 0x82b433f971cB72f84EeAB505CDD0b63F8445c346 |
| Buy Back Wallet | 0x914FaCbcDD285505C93E76816546d77b4c265B30 |
| Buy Back Burn Fee Receiver | 0xd9e6C8D9abDB4434F703C60D17eA6482ab3460ec |
| Bonus Fee Receiver | 0x735f87b751A7E13f0a19394D23D4d0B58419ea43 |
| Sale Factory Contract | 0xFEFC1b487bE52A9ED92e43C224764a647C8Eb38c |
| Presale Factory Contract | 0x5c75f975A42D2750FF6f0ecD9441b518Fdb06164 |
| Fair Launch Factory Contract | 0x09B4717bA7A433C4E2b8fa9748aA3080b0E9B5F5 |
| Contracts Owner | 0x26C7DaE7378929d2A94F74cCE3139876D9c21CF4 |
| Smart Contracts Creator | 0x3332F42de35e8614416E9852983751C9cA537847 |
| Blockchain | BSC Smart Chain Mainnet |



Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts

Checklist

| | |
|--|--------|
| Compiler errors. | Passed |
| Possible delays in data delivery. | Passed |
| Timestamp dependence. | Passed |
| Integer Overflow and Underflow. | Passed |
| Race Conditions and Reentrancy. | Passed |
| DoS with Revert. | Passed |
| DoS with block gas limit. | Passed |
| Methods execution permissions. | Passed |
| Economy model of the contract. | Passed |
| Private user data leaks. | Passed |
| Malicious Events Log. | Passed |
| Scoping and Declarations. | Passed |
| Uninitialized storage pointers. | Passed |
| Arithmetic accuracy. | Passed |
| Design Logic. | Passed |
| Impact of the exchange rate. | Passed |
| Oracle Calls. | Passed |
| Cross-function race conditions. | Passed |
| Fallback function security. | Passed |
| Safe Open Zeppelin contracts and implementation usage. | Passed |



Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts

| | |
|--|-------------|
| Whitepaper-Website-Contract correlation. | Not Checked |
| Front Running. | Not Checked |



Testing Summary

PASS

BLOCK SOLUTIONS BELIEVES

*This smart contracts passes the
security qualifications.*

15th January, 2025





Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts

Quick Stats:

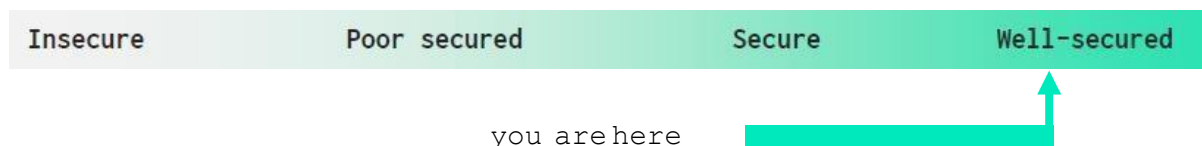
| Main Category | Subcategory | Result |
|----------------------|---|--------|
| Contract Programming | Solidity version not specified | Passed |
| | Solidity version too old | Passed |
| | Integer overflow/underflow | Passed |
| | Function input parameters lack of check | Passed |
| | Function input parameters check bypass | Passed |
| | Function access control lacks management | Passed |
| | Critical operation lacks event log | Passed |
| | Human/contract checks bypass | Passed |
| | Random number generation/use vulnerability | Passed |
| | Fallback function misuse | Passed |
| | Race condition | Passed |
| | Logical vulnerability | Passed |
| | Other programming issues | Passed |
| Code Specification | Visibility not explicitly declared | Passed |
| | Var. storage location not explicitly declared | Passed |
| | Use keywords/functions to be deprecated | Passed |
| | Other code specification issues | Passed |
| Gas Optimization | Assert () misuse | Passed |
| | High consumption 'for/while' loop | Passed |
| | High consumption 'storage' storage | Passed |
| | "Out of Gas" Attack | Passed |
| Business Risk | The maximum limit for mintage not set | Passed |
| | "Short Address" Attack | Passed |
| | "Double Spend" Attack | Passed |



Overall Audit Result: **Passed**

Executive Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **Well-Secured**. Again, it is recommended to perform an Extensive audit assessment to bring a more assured conclusion.



We used various tools like Mythril, Slither and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.

All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Quick Stat section.

We found critical, 0 high, 0 medium and 0 low level issues.

Code Quality

The SALE, PRESALE & FAIR LAUNCH FACTORY Smart Contracts protocol consists of three smart contracts. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in protocol. The BLOCKSOLUTIONS team has **not** provided scenario and unit test scripts, which would help to determine the integrity of the code in an automated way. Overall, the code is not commented. Commenting can provide rich documentation for functions, return variables and more.

Documentation

As mentioned above, it's recommended to write comments in the smart contract code, so anyone can quickly understand the programming flow as well as complex code logic. We were given a SALE, PRESALE & FAIR LAUNCH FACTORY Smart Contracts code in the form of File.

Use of Dependencies

As per our observation, the libraries are used in this smart contract infrastructure that are based on well-known industry standard open-source projects. And even core code blocks are written well and systematically. This smart contract does not interact with other external smart contracts.



Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts

| Risk Level | Description |
|--|--|
| Critical | Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc. |
| High | High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions |
| Medium | Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose |
| Low | Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution |
| Lowest / Code Style / Best Practice | Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored. |

Audit Findings

Critical

No Critical severity vulnerabilities were found.

High

No high severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

No Low severity vulnerabilities were found.



Conclusion

The Smart Contracts code passed the audit successfully. There were no warnings raised. We were given 3 contract code. And we have used all possible tests based on given objects as files. So, it is good to go for production. Since possible test cases can be unlimited for such extensive smart contract protocol, hence we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything. Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in Quick Stat section of the report. Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contracts is “Well Secured”.

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high-level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We



Smart Contract Code Review and Security Analysis Report for Sale, Presale & Fair Launch Factory Smart Contracts

generally, follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.