Security Audits

smart Contract Audits - KYC
Blockchain Security

$AFPEP TOKEN

AUDIT

SECURITY ASSESSMENT

27 March , 2025

FOR

Ancient FirePepe

Project Overview

Summary

Project Name	$AFPEP
Website	https://firepepe.com
About the project	FirePepe, Often seen as a role model, Fire Pepe has transcended its digital roots to symbolize a broader cultural and historical revolution. Fire Pepe emerged from the depths of ancient civilizations and, like a glowing fire, became a symbol of resilience, creativity, and the transformative power of online communities
Chain	Ethereum Network
Language	Solidity
Codebase	https://etherscan.io/address/0x5F53bCc29364C4A0796b2641c5ec6c0397f9c76B#code
Commit	N/A
Unit Tests	Not Provided

Social Medias

Telegram	https://t.me/AFPEPtoken
Twitter	https://x.com/AFPEPtoken
Facebook	N/A
Instagram	N/A
GitHub	N/A
Reddit	N/A
Medium	N/A
Discord	N/A
YouTube	N/A
TikTok	N/A
LinkedIn	N/A

Audit Summary

Version	Delivery Date	Change Log
v1.0	.27 March 2025	· Layout Project · Automated/Manual-Security Testing · Summary

Note – The following audit report presents a comprehensive security analysis of the smart contract utilized in the project that includes outside manipulation of the contract's functions in a malicious way. This analysis did not include functional testing (or unit testing) of the contract/s logic. We cannot guarantee 100% logical correctness of the contract as we did not functionally test it. This includes internal calculations in the formulae used in the contract.

File Overview

The Team provided us with the files that should be tested in the security assessment. This audit covered the following files listed below with an SHA-1 Hash.

File Name	SHA-1 Hash
contracts/AFPEP.sol	a18ea332400de7ca8a5696a70570c9c851badc0

Please note: Files with a different hash value than in this table have been modified after the security check, either intentionally or unintentionally. A different hash value may (but need not) be an indication of a changed state or potential vulnerability that was not the subject of this scan.

Imported packages.

Used code from other Frameworks/Smart Contracts.

N/A

Note for Investors: We only audited contracts mentioned in the scope above. All contracts related to the project apart from that are not a part of the audit, and we cannot comment on its security and are not responsible for it in any way.

External/Public functions

External/public functions are functions that can be called from outside of a contract, i.e., they can be accessed by other contracts or external accounts on the blockchain. These functions are specified using the function declaration's external or public visibility modifier.

State variables

State variables are variables that are stored on the blockchain as part of the contract'sstate. They are declared at the contract level and can be accessed and modified by any function within the contract. State variables can be needed within visibility modifier, such as public, private or internal, which determines the access level of the variable.

Components

📝 Contracts	📚 Libraries	🔍 Interfaces	🎨 Abstract
1	0	3	0

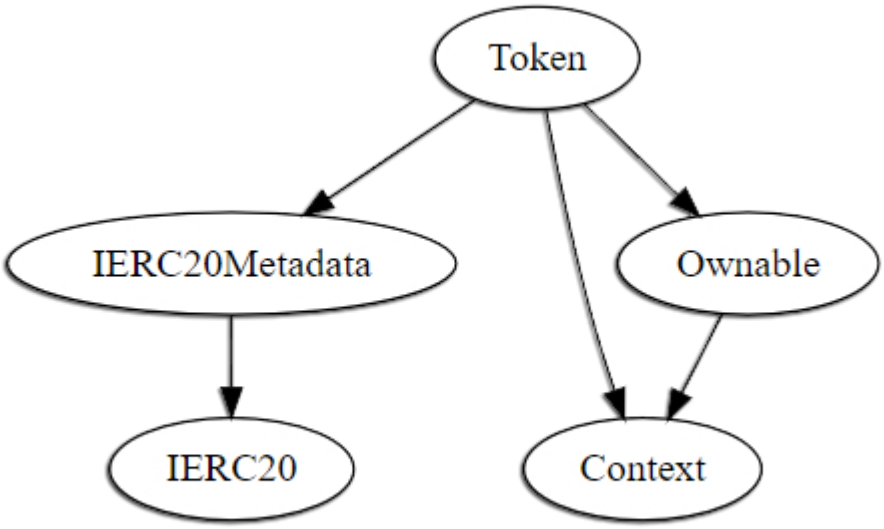
Capabilities

Solidity Versions observed	Experimental Features	Can Receive Funds	📟 Uses Assembly	Has Destroyable Contracts
`0.8.28`	`----------`	`Yes`		`----------`

Transfers ETH	Low-Level Calls	Delegate Call	Uses Hash Functions	ECRecover	New/Create/Create2
`yes`					

Inheritance Graph

An inheritance graph is a graphical representation of the inheritance hierarchy among contracts. In object-oriented programming, inheritance is a mechanism that allows one class (or contract, in the case of Solidity) to inherit properties and methodsfrom another class. It shows the relationships between different contracts and how they are related to each other through inheritance.



Audit Information

Vulnerability & Risk Level

Risk represents the probability that a certain source threat will exploit thevulnerability and the impact of that event on the organization or system.The risk level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executingthe contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to check the repository for security-related issues, code quality, and compliance with specifications and best practices. To this end, our team of experienced pen-testers andsmart contract developers reviewed the code line by line and documented any issues discovered.

We check every file manually. We use automated tools only so that they help us achieve faster and better results.

Methodology

The auditing process follows a routine series of steps:

1. Codereviewthat includes the following :

2. Reviewingthe specifications, sources, and instructions provided to ensure w understand the size, scope, and functionality of the smart contract.

a. Manualreview of the code, i.e., reading the source code line by line to identify potential vulnerabilities.

b. Comparisont o the specification, i.e., verifying that the code does what is described in the specifications, sources, and instructions provided.

3. Testingandautomated analysis that includes the following :

a. Testcoverage analysis determines whether test cases cover code and how much code is executed when those test cases are executed.

b. Symbolicexecution, which is analysing a program to determine what inputs cause each part of a program t o execute.

4. Reviewbest practices, i.e., review smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on best practices, recommendations, and research from industry and academia.

5. Concrete,itemized and actionable recommendations t o help you secure your smart contract.

Overall Security

Upgradeability

Contract is not an upgradable	✅ **Deployer cannot update the contract with new functionalities.**
Description	The contract is not an upgradeable contract. The Deployer is not able to change or add any functionalities to the contract after deploying.
Comment	N/A

Ownership

Contract ownership is renounced.	✅ **The ownership is renounced.**

Description	There are no ownership privileges in this contract.
Comment	N/A

Note – *The contract cannot be considered as renounced till it is not deployed or having some functionality that can change the state of the contract.*

Ownership Privileges

These functions can be dangerous. Please note that abuse can lead to financial loss. We have a guide where you can learn more about these Functions.

Minting tokens

Minting tokens refer to the process of creating new tokens in a cryptocurrency or blockchain network. This process is typically performed by the project's owner or designated authority, who has the ability to add new tokens to the network's total supply.

Contract owner cannot mint new tokens.	✅ **The owner cannot mint new tokens.**

Description	The owner is not able to mint new tokens once the contract is deployed.
Comment	N/A

Burning tokens

Burning tokens is the process of permanently destroying a certain number of tokens, reducing the total supply of a cryptocurrency or token. This is usually done to increasethe value of the remaining tokens, as the reduced supply can create scarcity and potentially drive up demand.

Contract owner cannot burn tokens	✅ **The owner cannot burn tokens.**
Description	The owner is not able burn tokens without any allowances.
Comment	N/A

Blacklist addresses

Blacklisting addresses in smart contracts is the process of adding a certain address to a blacklist, effectively preventing them from accessingor participating in certain functionalities or transactions within the contract. This can be useful in preventing fraudulent or malicious activities, such as hacking attempts or money laundering.

Contract owner cannot blacklist addresses.	✅ **The owner cannot blacklist wallets.**
Description	The owner cannot blacklist wallets from transferring tokens.
Comment	N/A

Fees and Tax

In some smart contracts, the owner or creator of the contract can setfees for certain actions or operations within the contract. These fees can be used to cover the cost of running the contract, such as paying for gas fees or compensating the contract's owner for their time and effort indeveloping and maintaining the contract.

Contract owner cannot .set fees more than 15%	✅ **.The owner cannot set fees more than 15%**

Description	.The owner cannot set fees of more than 15%
Comment	N/A

Lock User Funds

In a smart contract, locking refers to the process of restricting access to certain tokens or assets for a specified period of time. When token or assets are locked in a smart contract, they cannot be transferred or used until the lock-up period has expired or certain conditions have been met.

Contract owner cannot lock function.	✅ **The owner cannot lock function.**
Description	The owner cannot lock the contract.
Comment	N/A

Centralization Privileges

Centralization can arise when one or more parties have privileged access or control over the contract's functionality, data, or decision-making. This can occur, for example,if the contract is controlled by a single entity or if certain participants have special permissions or abilities that others do not.

In the project, there are authorities that have access to the following functions:

File	Privileges
AFPEP.sol	➢ There are no ownership privileges in the contract. The owner cannot change any settings in the contract.

In this contract, roles are clearly and functionally defined, which improves the security and performance of the contract. Granting roles .has been very useful for transparency and decentralization

Audit Result

Critical Issues

No critical issues

High Issues

No high issues

Medium Issue

No medium issues

Low Issue

No low issues

Informational Issue

#1 | Functions that are not used.

File	Severity	Location	Status
AFPEP.sol	Informational	--	Open

.**Description** – Unused functions were removed

Legend for the Issue Status

Attribute or Symbol	Meaning
Open	The issue is not fixed by the project team.
Fixed	The issue is fixed by the project team.
Acknowledged(ACK)	The issue has been acknowledged or declared as part of business logic.