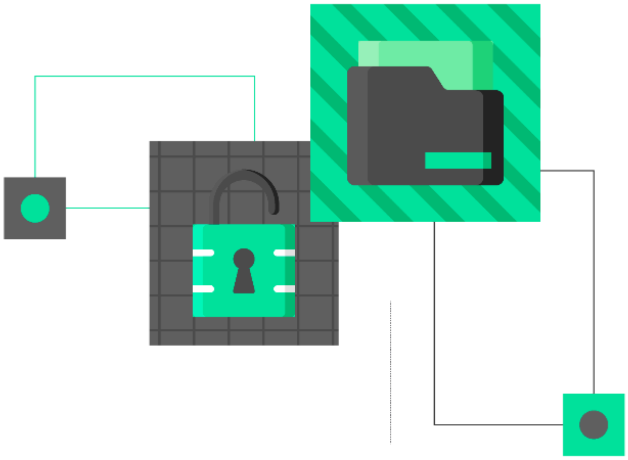
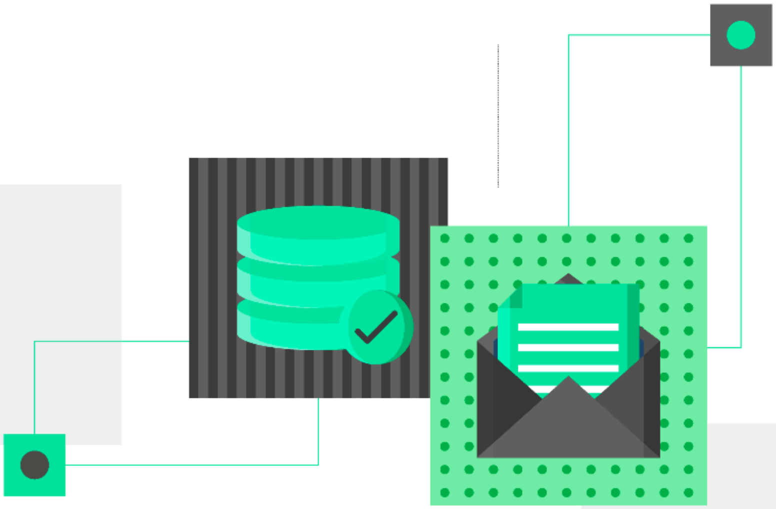
Mantisec Labs

Smart Contract Audit



Presearch

PRE-Token-Base



Apr 2024

Contents

Disclaimer

This disclaimer is to inform you that the report you are reading has been prepared by Mantisec Labs for informational purposes only and should not be considered as investment advice. It is important to conduct your own independent investigation before making any decisions based on the information contained in the report. The report is provided "as is" without any warranties or conditions of any kind and Mantisec Labs excludes all representations, warranties, conditions, and other terms. Additionally, Mantisec Labs assumes no liability or responsibility for any kind of loss or damage that may result from the use of this report.

It is important to note that the analysis in the report is limited to the security of the smart contracts only and no applications or operations were reviewed. The report contains proprietary information, and Mantisec Labs holds the copyright to the text, images, photographs, and other content. If you choose to share or use any part of the report, you must provide a direct link to the original document and credit Mantisec Labs as the author.

By reading this report, you are accepting the terms of this disclaimer. If you do not agree with these terms, it is advisable to discontinue reading the report and delete any copies in your possession.

Audit Process & Methodology

The Mantisec Labs team carried out a thorough audit for the project, starting with an in-depth analysis of code design patterns. This initial step ensured the smart contract's architecture was well-structured and securely integrated with third-party smart contracts and libraries.Also, our team conducted a thorough line-by-line inspection of the smart contract,seeking out potential issues such as Signature Replay Attacks, Unchecked External Calls, External Contract Referencing, Variable Shadowing, Race conditions, Transaction-ordering dependence, timestamp dependence, DoS attacks, among others.

During the Unit testing phase, we assessed the functions authored by the developer to ascertain their precise functionality. Our Automated Testing procedures leveraged proprietary tools designed in-house to spot vulnerabilities and security flaws within the Smart Contract. The code was subjected to an in-depth audit administered by an independent team of auditors, encompassing the following critical aspects:

- Scrutiny of the smart contract's structural analysis to verify its integrity.
- Extensive automated testing of the contract
- A manual line-by-line Code review, undertaken with the aim of evaluating, analyzing, and identifying potential security risks.
- An evaluation of the contract's intended behavior, encompassing a review of provided documentation to ensure the contract conformed to expectations.
- Rigorous verification of storage layout in upgradeable contracts.
- An integral component of the audit procedure involved the identification and recommendation of enhanced gas optimization techniques for the contract

Audit Purpose

Mantisec Labs was hired by the Presearch team to review their smart contract. This audit was conducted in **April 2024**.

The main reasons for this review were:

- To find any possible security issues in the smart contract.
- To carefully check the logic behind the given smart contract.

This report provides valuable information for assessing the level of risk associated with this smart contract and offers suggestions on how to improve its security by addressing any identified issues.

Contract Details

Project Name	Presearch
Contract link	https://github.com/PresearchOfficial/PRE-Token-Base/tree/main/contracts
Language	Solidity
Type	ERC20

Security Level Reference

Each problem identified in this report has been categorized into one of the following severity levels:

- **High** severity issues pose significant risks and should be addressed promptly.
- **Medium** severity issues have the potential to create problems and should be on the agenda for future fixes.
- **Low** severity issues are minor concerns and warnings. While they may not require immediate action, addressing them in the future is advisable for overall improvement.

Issues	**High**	**Medium**	**Low**
Open	0	0	0
Closed	1	-	3

Findings

Contract Name: PRE-Token-Base

L001- Don't Initialize Variables with Default Value

If a variable is not set/initialized, it is assumed to have the default value (0, false, 0x0 etc depending on the data type). If you explicitly initialize it with its default value, you are just wasting gas.

Code Location:
~/PRE-Token-Base-main/contracts/PresearchCommonERC20.sol::102

L002- Cache Array Length Outside of Loop

Reading array length at each iteration of the loop takes 6 gas (3 for mload and 3 to place memory_offset) in the stack.

Code Location:
~/PRE-Token-Base-main/contracts/PresearchCommonERC20.sol::102

L003- Redundant Initialization Function Found in Contract

__PresearchCommonERC20_init() is calling __PresearchCommonERC20_init_unchained() within the same contract and has no reference, which seems unnecessary. Both functions have the same access modifier and purpose.

To streamline the code and remove redundancy, you can eliminate one of the functions and directly call __PresearchCommonERC20_init_unchained() from initialize()

H001- Incorrect Authorization State Check

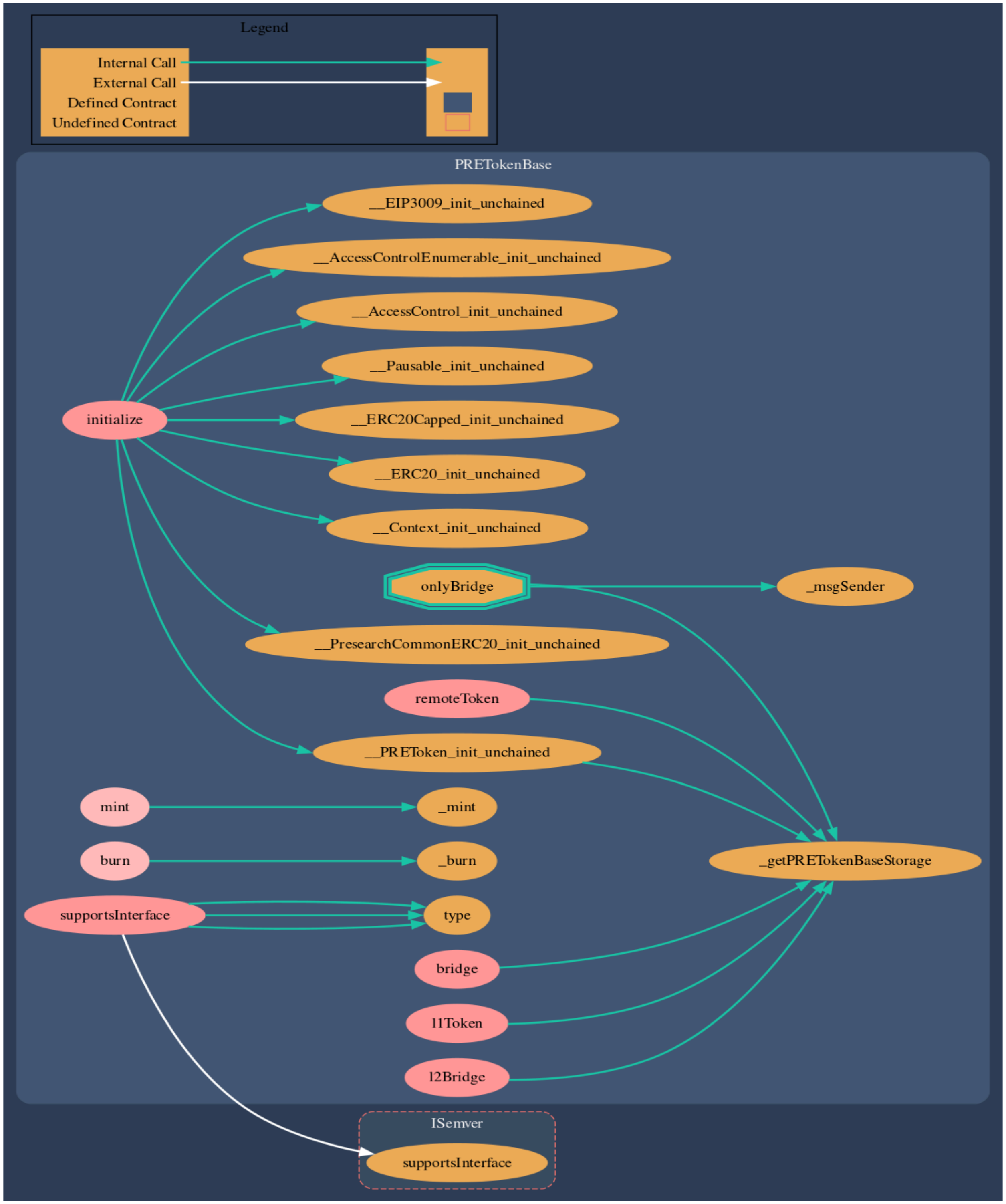
The bug occurs in the c̀ancelAuthorizatioǹ function of the contract ÈIP3009.sol̀. In this function, the validation logic for checking whether an authorization has been used.

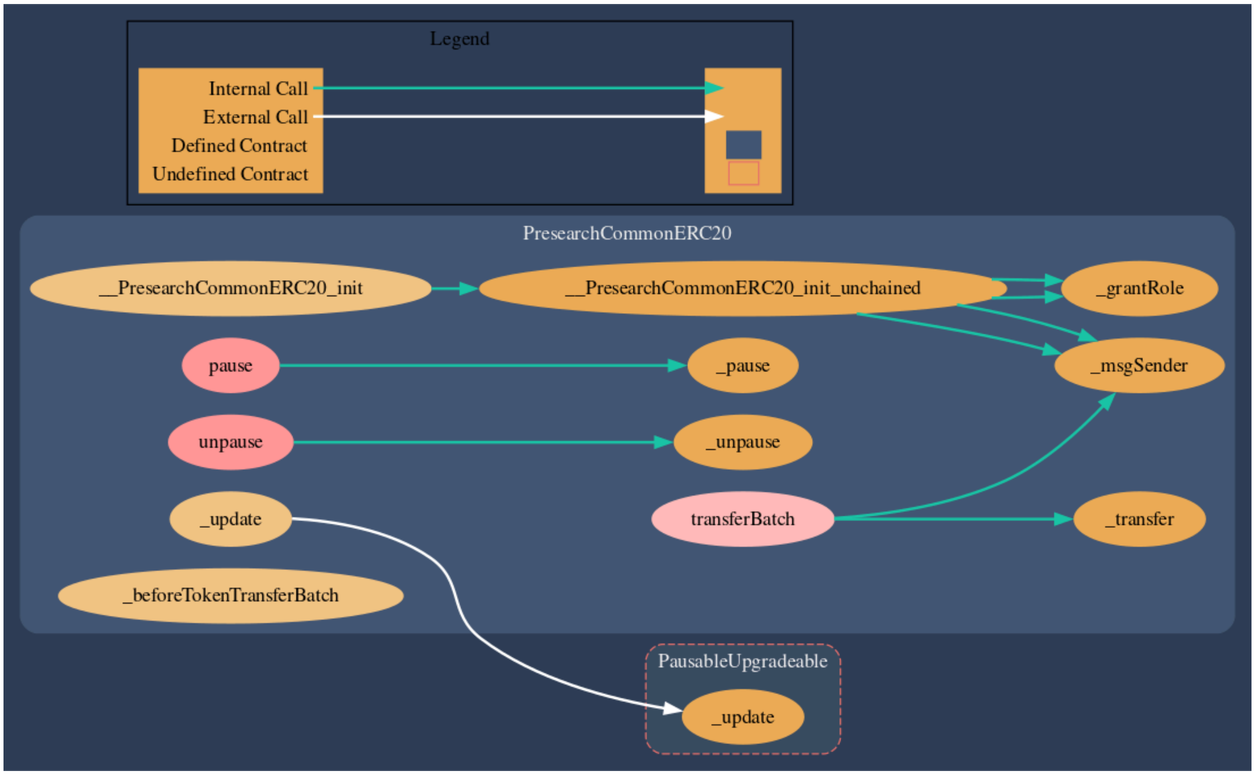
Currently, the function erroneously allows the cancellation of authorizations that have already been used.The incorrect validation logic poses a security risk to the integrity and functionality of the contract.To address this issue, the validation logic in the c̀ancelAuthorizatioǹ function needs to be corrected.

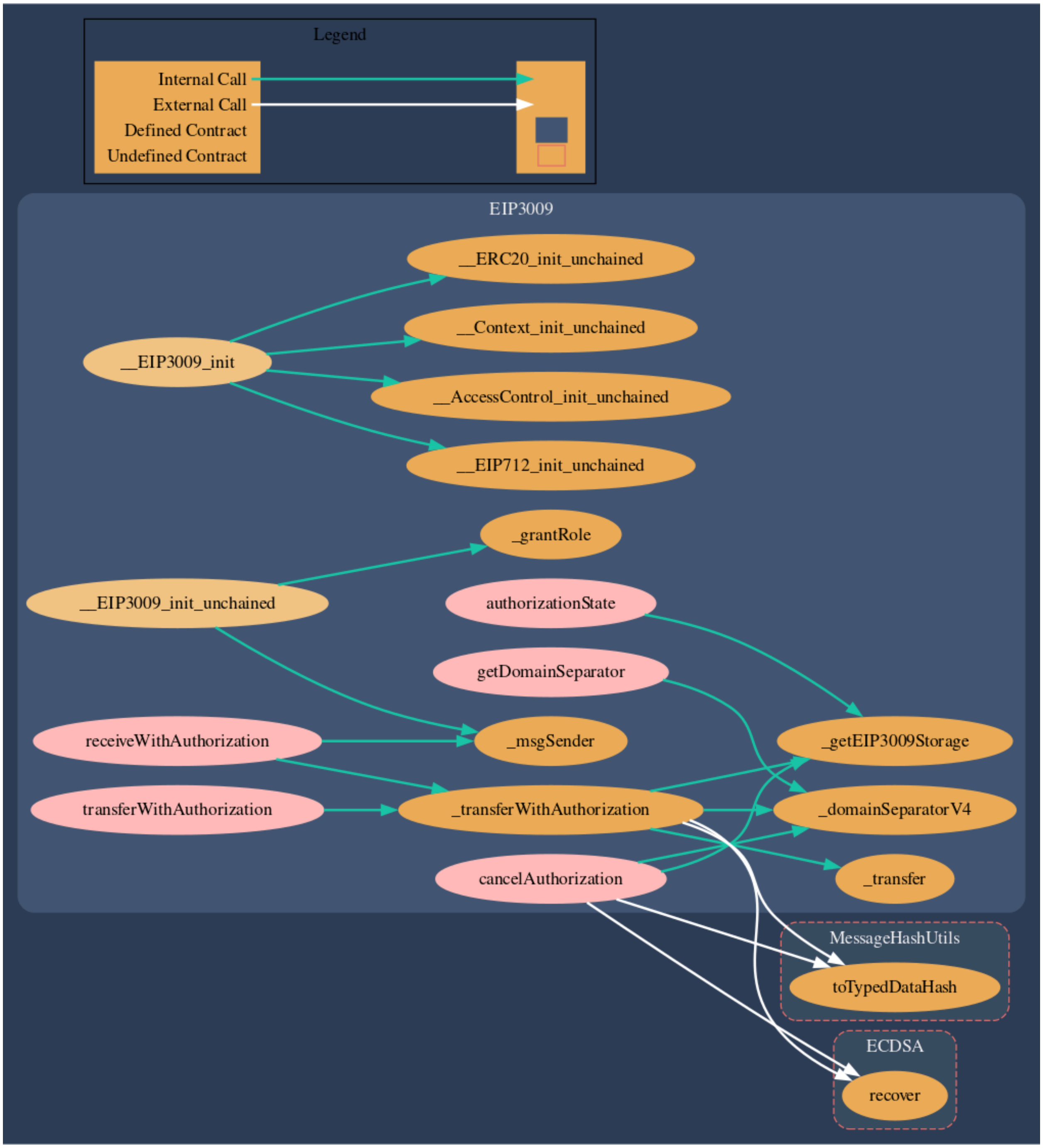
It should be this:

```
require(
           !$._authorizationStates[authorizer][nonce],
           _AUTHORIZATION_USED_ERROR
       );
```

Additional Details




Legend
Internal Call
External Call
Defined Contract
Undefined Contract
PresearchCommonERC20
__PresearchCommonERC20_init
__PresearchCommonERC20_init_unchained
_grantRole
pause
_pause
_msgSender
unpause
_unpause
_update
transferBatch
_transfer
_beforeTokenTransferBatch
PausableUpgradeable
_update


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
EIP3009
__ERC20_init_unchained
__Context_init_unchained
__EIP3009_init
__AccessControl_init_unchained
__EIP712_init_unchained
_grantRole
__EIP3009_init_unchained
authorizationState
getDomainSeparator
receiveWithAuthorization
_msgSender
_getEIP3009Storage
transferWithAuthorization
_transferWithAuthorization
_domainSeparatorV4
cancelAuthorization
_transfer
MessageHashUtils
toTypedDataHash
ECDSA
recover

Concluding Remarks

To wrap it up, this PRE-Token-Base audit has given us a good look at the contract's security and functionality.
Our auditors confirmed that all the issues are now resolved by the developers.