Smart Contract Security Audit V1

XGrowth Smart Contract Audit

https://xfuzion.finance/

Feb 26, 2024

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

Table of Contents

Table of Contents

Background

Project Information

Smart Contract Information

Executive Summary

File and Function Level Report

File in Scope:

Issues Checking Status

SWC Attack Analysis

Severity Definitions

Audit Findings

Automatic testing

Testing proves

Inheritance graph

Call graph

Source lines

Risk level

Source units in scope

Capabilities

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

Background

The purpose of the audit was to achieve the following:

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

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

Project Information

- **Platform**: Pulse Chain
- **Name**: XGrowth
- **Language** : solidity
- **Contract Address**: 0xeeBC5a331FEE9a7f30D75B32E9E096B7171295C2
- **Code Source**: https://scan.pulsechain.com/address/0xeeBC5a331FEE9a7f30D75B32E9E096B7171295C2
- **Website**: https://xfuzion.finance/
- **Telegram**: https://t.me/XfuzionFinance
- **X**: https://x.com/xfuzionfinance?s=21&t=UppJadbfgCRdnVpOnFAakQ
- **Youtube:** https://youtube.com/@XfuzionFinance?si=nfNwrOVxVWNJgnq2
- **Discord**: https://discord.gg/FCcD8YTuxc

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Well-Secured**.

Well Secured	✓
Secured	
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 2 low, 0 very low-level issues and 2 note in all solidity files of the contract

The files:

XGrowth.sol

Audit Score:

99% secure

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
XGrowth.sol	a007d1f2cbace3fd88435141789baa2be2b980d2	0xeeBC5a331FEE9a7f30D75B32E9E096B7171295C2

- Contract: XGrowth
- Inherit: Ownable, ReentrancyGuard
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
_currentDepositID	✓	Read / public	**Passed**
apr	✓	Read / public	**Passed**
buyBurnAddress	✓	Read / public	**Passed**
buyBurnFeeBP	✓	Read / public	**Passed**
daiToken	✓	Read / public	**Passed**
depositFeeBP	✓	Read / public	**Passed**
devFeeAddress	✓	Read / public	**Passed**
devFeeBP	✓	Read / public	**Passed**
getBalance	✓	Read / public	**Passed**
investors	✓	Read / public	**Passed**
owner	✓	Read / public	**Passed**
lendXAddress	✓	Read / public	**Passed**
lendXFeeBP	✓	Read / public	**Passed**
minAmount	✓	Read / public	**Passed**

minClaimAmount	✓	Read / public	**Passed**
operator	✓	Read / public	**Passed**
ownedDeposits	✓	Read / public	**Passed**
pendingReward	✓	Read / public	**Passed**
rewardPeriod	✓	Read / public	**Passed**
totalDepositLimit	✓	Read / public	**Passed**
totalInvested	✓	Read / public	**Passed**
totalInvestors	✓	Read / public	**Passed**
totalReward	✓	Read / public	**Passed**
claimAllReward	✓	Write / public	**Passed**
deposit	✓	Write / public	**Passed**
setDevFeeAddress	✓	Write / public	**Passed**
setDevFeeBP	✓	Write / public	**Passed**
setLendXAddress	✓	Write / public	**Passed**
setLendXFee	✓	Write / public	**Passed**
updateApr	✓	Write / public	**Passed**
withdrawFundsForExternal	✓	Write / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**

Issues Checking Status

SWC Attack Analysis

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) for more info check https://swcregistry.io/

No.	Issue Description	Checking Status
136	Unencrypted Private Data On-Chain	**Passed**
135	Code With No Effects	**Passed**
134	Message call with hardcoded gas amount	**Passed**
133	Hash Collisions With Multiple Variable Length Arguments	**Passed**
132	Unexpected Ether balance	**Passed**
131	Presence of unused variables	**Passed**
130	Right-To-Left-Override control character (U+202E)	**Passed**
129	Typographical Error	**Passed**
128	DoS with block gas limit.	**Passed**
127	Arbitrary Jump with Function Type Variable	**Passed**
126	Insufficient Gas Griefing	**Passed**
125	Incorrect Inheritance Order	**Passed**
124	Write to Arbitrary Storage Location	**Passed**
123	Requirement Violation	**Passed**
122	Lack of Proper Signature Verification	**Passed**
121	Missing Protection against Signature Replay Attacks	**Passed**
120	Weak Sources of Randomness from Chain Attributes	**Passed**

119	Shadowing State Variables	**Passed**
118	Incorrect Constructor Name	**Passed**
117	Signature Malleability	**Passed**
116	Block values as a proxy for time	**Not Passed**
115	Authorization through tx.origin	**Passed**
114	Transaction Order Dependence	**Passed**
113	DoS with Failed Call	**Passed**
112	Delegatecall to Untrusted Callee	**Passed**
111	Use of Deprecated Solidity Functions	**Passed**
110	Assert Violation	**Passed**
109	Uninitialized Storage Pointer	**Passed**
108	State Variable Default Visibility	**Passed**
107	Reentrancy	**Passed**
106	Unprotected SELFDESTRUCT Instruction	**Passed**
105	Unprotected Ether Withdrawal	**Passed**
104	Unchecked Call Return Value	**Passed**
103	Floating Pragma	**Passed**
102	Outdated Compiler Version	**Not Passed**
101	Integer Overflow and Underflow	**Passed**
100	Function Default Visibility	**Passed**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens 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
Note	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:

#Missing zero address validation

When the owner wants add the Dev fee address, he has to check for the zero address to make, he didn't add the zero address. Otherwise, he will lose the fee.

```
function setDevFeeAddress(address _devFeeAddress) external onlyOwner{
        devFeeAddress = _devFeeAddress;
    }
```

Remediation

Use the require statement to check for zero addresses.

Status: Acknowledged.

Use of block.timestamp for comparisons

The value of block.timestamp can be manipulated by the miner. And conditions with strict equality is difficult to achieve - block.timestamp.

```
function pendingReward(address _user) public view returns (uint256) {
        if(ownedDeposits[_user].length == 0) return 0;

        uint256 currentTime = block.timestamp;
        uint256 lastClaimedDate = investors[_user].lastClaimedAt;
        uint256 _totalReward = 0;
        for(uint i = 0; i < ownedDeposits[_user].length; i++) {
            if(depositStates[ownedDeposits[_user][i]].depositAt + rewardPeriod <
currentTime) {
                currentTime = depositStates[ownedDeposits[_user][i]].depositAt +
rewardPeriod;
            }
            _totalReward += depositStates[ownedDeposits[_user][i]].depositAmount *
apr * (currentTime - lastClaimedDate) / 86400 / 10000;
```

```
        }

        return _totalReward;
    }
```

Recommendation

Avoid use of block.timestamp.

Status

Acknowledged.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Compiler version is old

Description

The compiler being used was released 3 years ago. It's recommended to use more recent compiler version, there can be benefits like reduction in bytecode size etc.

Status: Acknowledged.

#Naming Conventions

Description

The contract follows a consistent naming convention where we are private variables with leading"_" and public variables without it. But we have missed to comply to the condition for certain variable names "__ _currentDepositID" which is public.

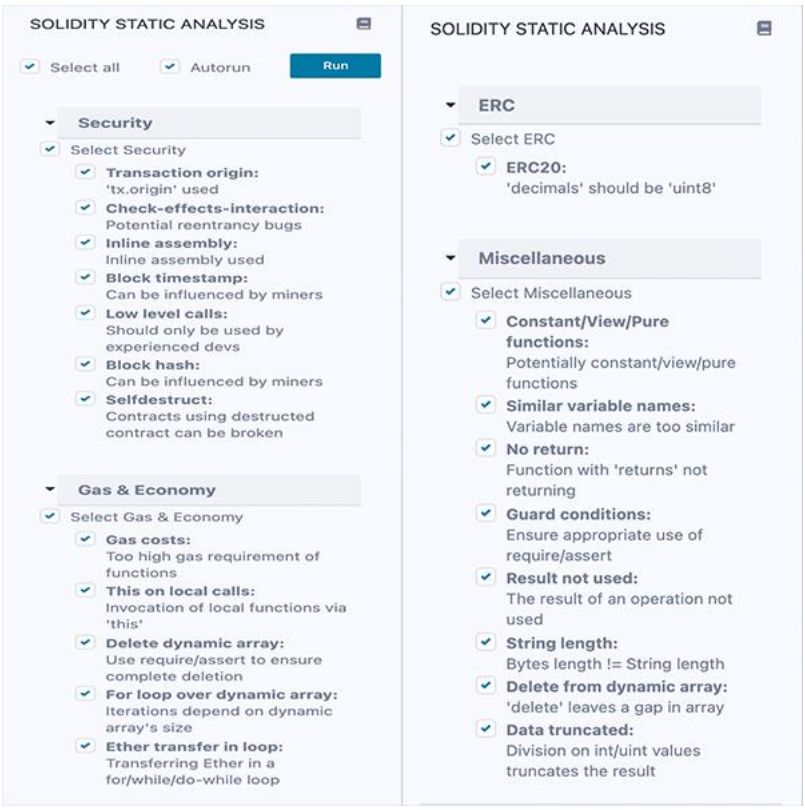
Remediation

Remove "_" from external variable names and add it to private variable names.

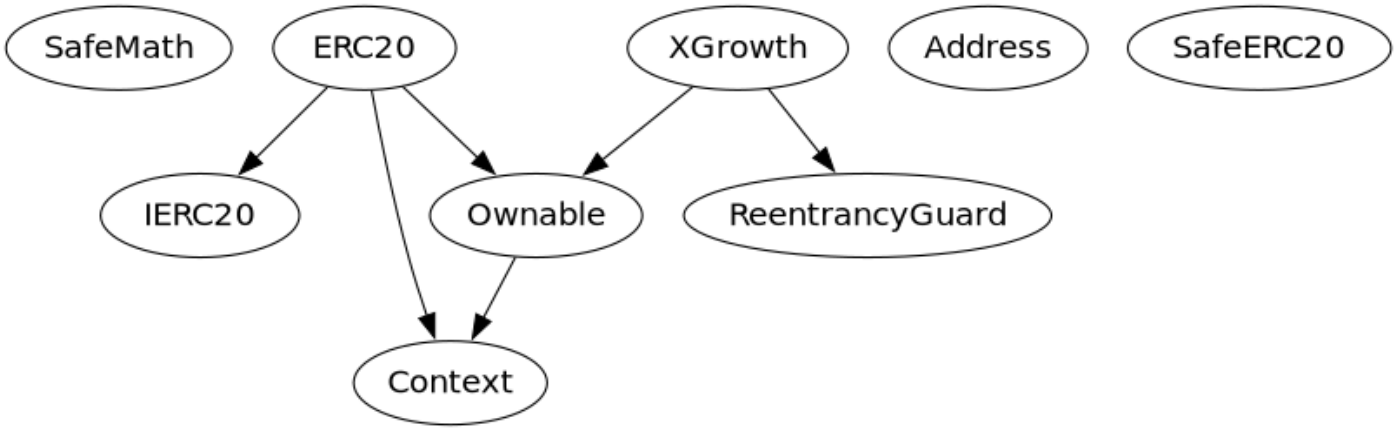
Status: Acknowledged.

Automatic Testing

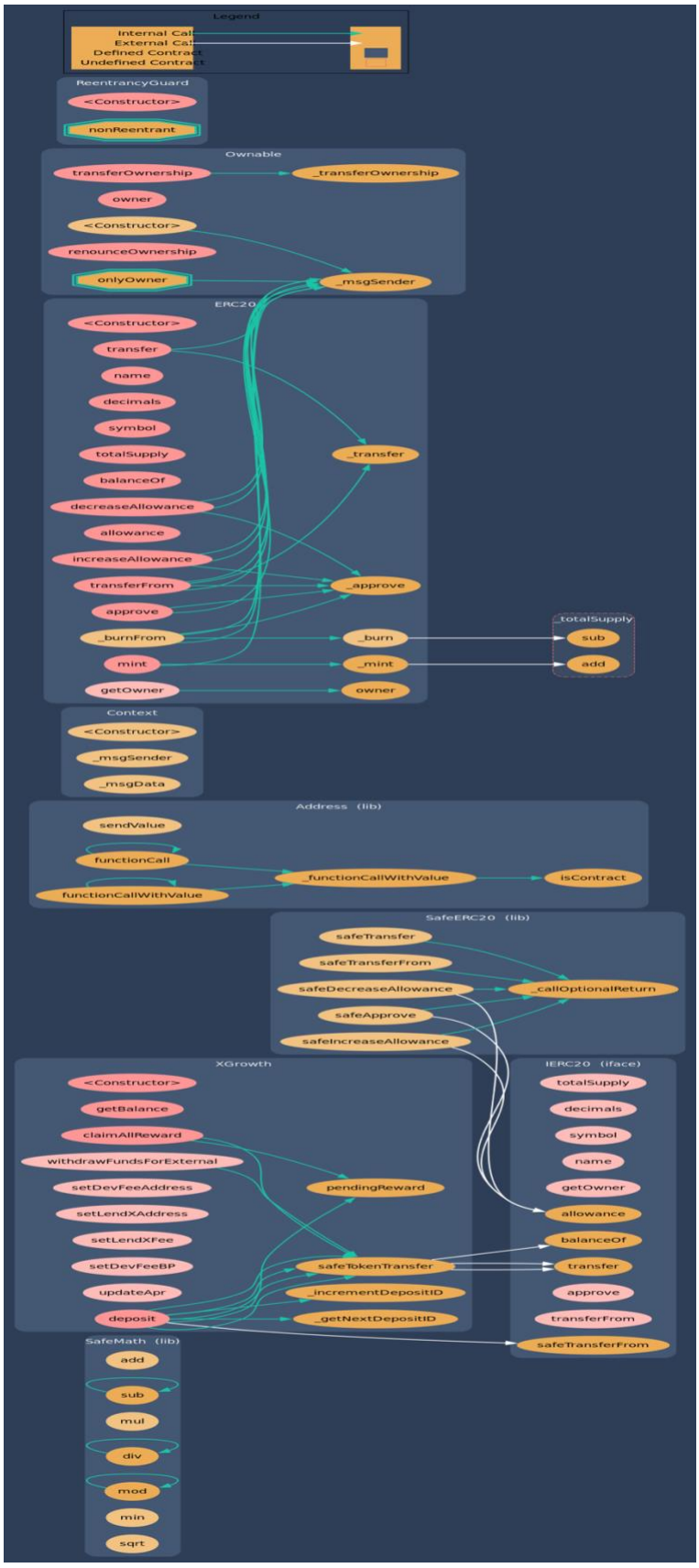
1- SOLIDITY STATIC ANALYSIS



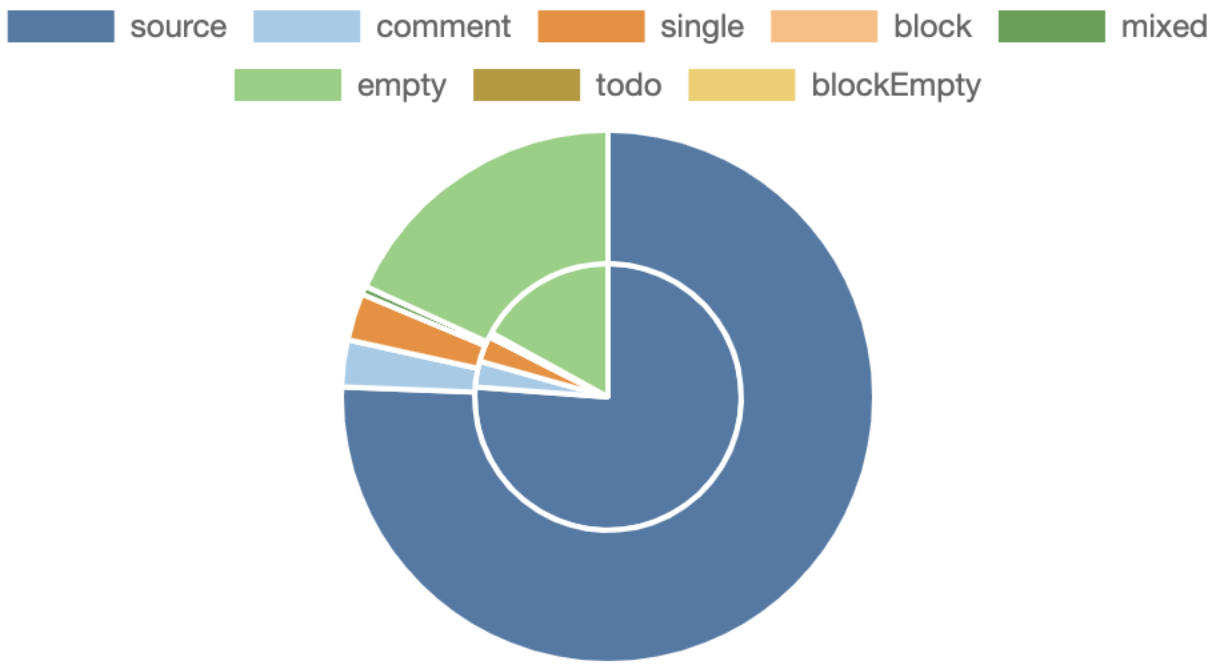
2- Inheritance graph



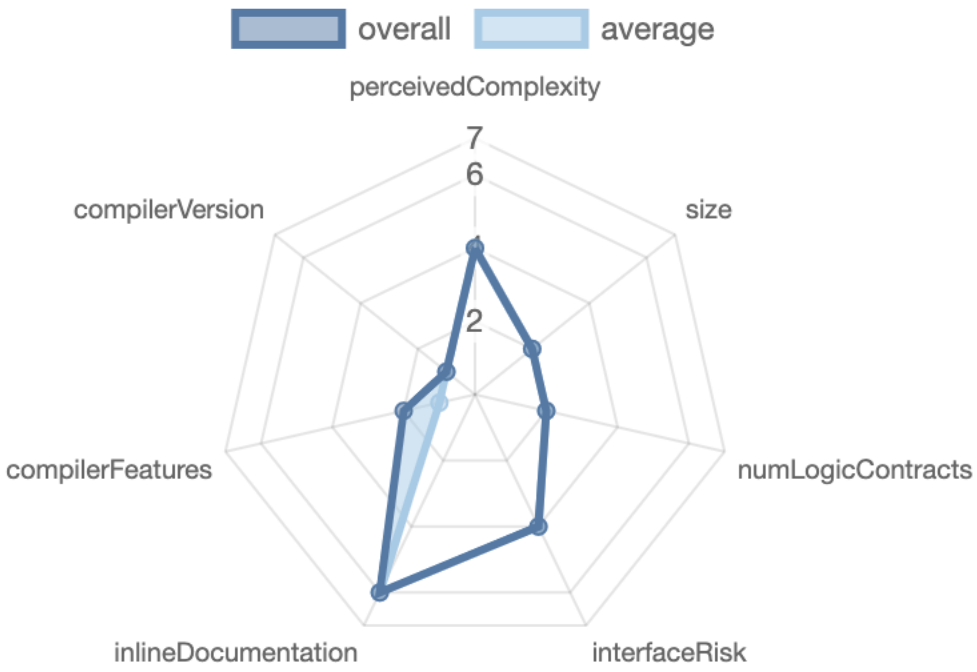
3- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
ReentrancyGuard
<Constructor>
nonReentrant
Ownable
transferOwnership
_transferOwnership
owner
<Constructor>
renounceOwnership
onlyOwner
_msgSender
ERC20
<Constructor>
transfer
name
decimals
symbol
totalSupply
balanceOf
decreaseAllowance
allowance
increaseAllowance
transferFrom
approve
_burnFrom
mint
getOwner
_transfer
_approve
_burn
_mint
owner
totalSupply
sub
add
Context
<Constructor>
_msgSender
_msgData
Address (lib)
sendValue
functionCall
functionCallWithValue
_functionCallWithValue
isContract
SafeERC20 (lib)
safeTransfer
safeTransferFrom
safeDecreaseAllowance
safeApprove
safeIncreaseAllowance
_callOptionalReturn
XGrowth
<Constructor>
getBalance
claimAllReward
withdrawFundsForExternal
setDevFeeAddress
setLendXAddress
setLendXFee
setDevFeeBP
updateApr
deposit
pendingReward
safeTokenTransfer
_incrementDepositID
_getNextDepositID
IERC20 (iface)
totalSupply
decimals
symbol
name
getOwner
allowance
balanceOf
transfer
approve
transferFrom
safeTransferFrom
SafeMath (lib)
add
sub
mul
div
mod
min
sqrt

Source lines



Risk level



Source units in scope

Source Units in Scope

Source Units Analyzed: `1`
Source Units in Scope: `1` **(100%)**

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
📝📚🔍🎨	XGrowth.sol	8	1	571	519	413	17	315	💻📩🔆
📝📚🔍🎨	**Totals**	**8**	**1**	**571**	**519**	**413**	**17**	**315**	💻📩🔆

Legend: [➖]

- **Lines**: total lines of the source unit
- **nLines**: normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC**: normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines**: lines containing single or block comments
- **Complexity Score**: a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

Capabilities

Components

📝Contracts	📚Libraries	🔍Interfaces	🎨Abstract
4	3	1	1

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

🌐Public	💰Payable
39	0

External	Internal	Private	Pure	View
17	76	4	10	21

StateVariables

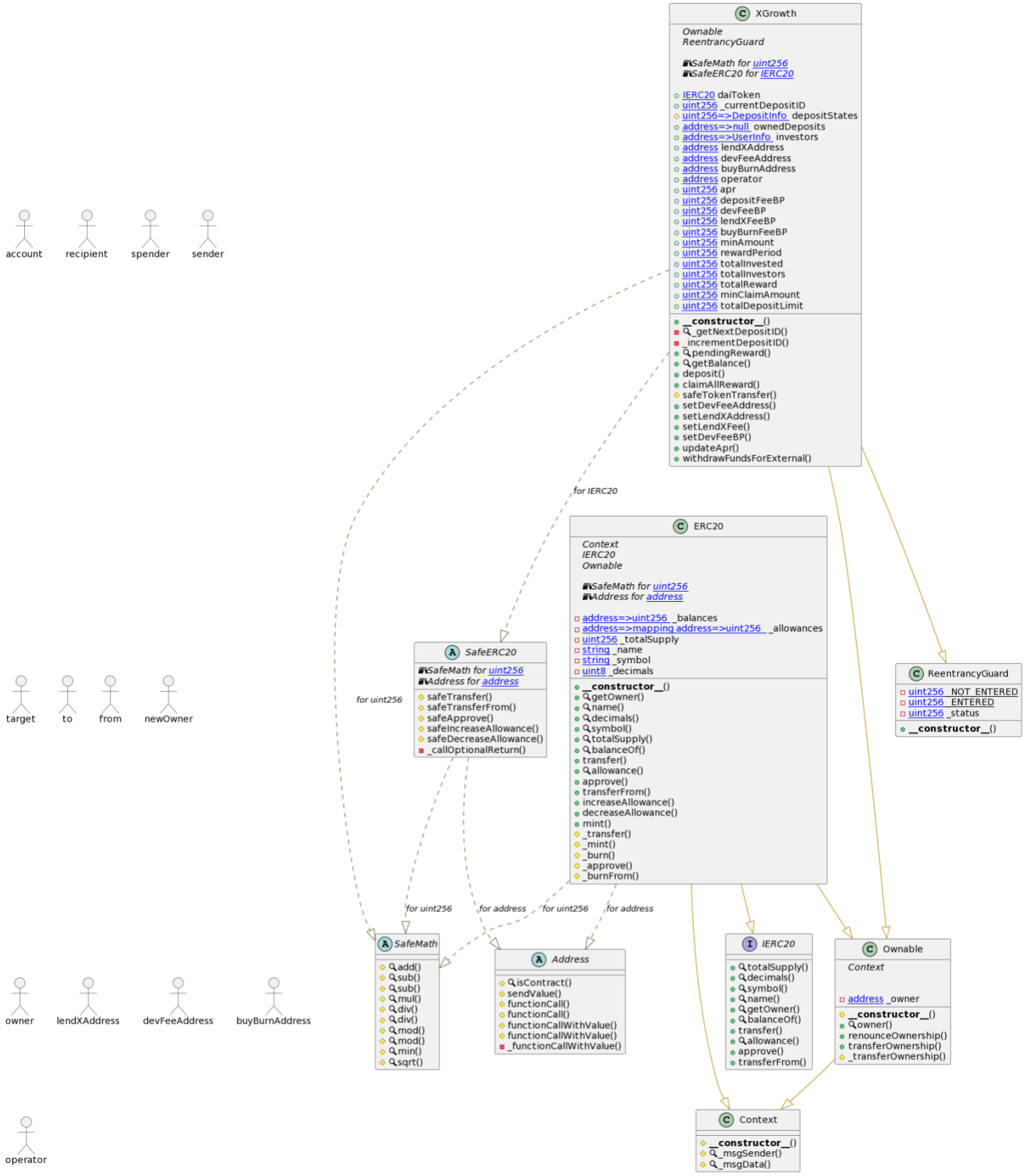
Total	🌐Public
31	20

Capabilities

Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥️ Uses Assembly	💣 Has Destroyable Contracts
`0.6.12`			`yes` (2 asm blocks)	

📩 Transfers ETH	⚡ Low-Level Calls	👥 DelegateCall	🧱 Uses Hash Functions	🍬 ECRecover	🌀 New/Create/Create2
`yes`					

Unified Modeling Language (UML)



Functions signature

Function Name	Sighash	Function Signature
totalSupply	18160ddd	totalSupply()
decimals	313ce567	decimals()
symbol	95d89b41	symbol()
name	06fdde03	name()
getOwner	893d20e8	getOwner()
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)
getOwner	893d20e8	getOwner()
name	06fdde03	name()
decimals	313ce567	decimals()
symbol	95d89b41	symbol()
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)
mint	a0712d68	mint(uint256)
pendingReward	f40f0f52	pendingReward(address)
getBalance	12065fe0	getBalance()
deposit	b6b55f25	deposit(uint256)
claimAllReward	8a623d86	claimAllReward()
setDevFeeAddress	d687b4d1	setDevFeeAddress(address)
setLendXAddress	263ea8f3	setLendXAddress(address)
setLendXFee	fcdd07a1	setLendXFee(uint256)
setDevFeeBP	b07a5939	setDevFeeBP(uint256)
updateApr	8552bf90	updateApr(uint256)
withdrawFundsForExternal	b0f71761	withdrawFundsForExternal(uint256)

Automatic general report

```
Files Description Table
|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/XGrowth.sol |
a007d1f2cbace3fd88435141789baa2be2b980d2 |
 Contracts Description Table
|  Contract  |         Type        |       Bases      |                  |
|
|:----------:|:-------------------:|:----------------:|:----------------:|:--------
-------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |
**Modifiers**  |
||||||
| **SafeMath** | Library |  |||
| └ | add | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | mul | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
| └ | mod | Internal 🔒 |   | |
| └ | mod | Internal 🔒 |   | |
| └ | min | Internal 🔒 |   | |
| └ | sqrt | Internal 🔒 |   | |
||||||
| **IERC20** | Interface |  |||
| └ | totalSupply | External ❗ |   |NO❗ |
| └ | decimals | External ❗ |   |NO❗ |
| └ | symbol | External ❗ |   |NO❗ |
| └ | name | External ❗ |   |NO❗ |
| └ | getOwner | External ❗ |   |NO❗ |
| └ | balanceOf | External ❗ |   |NO❗ |
| └ | transfer | External ❗ | 🛑 |NO❗ |
| └ | allowance | External ❗ |   |NO❗ |
| └ | approve | External ❗ | 🛑 |NO❗ |
| └ | transferFrom | External ❗ | 🛑 |NO❗ |
||||||
| **Address** | Library |  |||
| └ | isContract | Internal 🔒 |   | |
| └ | sendValue | Internal 🔒 | 🛑 | |
| └ | functionCall | Internal 🔒 | 🛑 | |
| └ | functionCall | Internal 🔒 | 🛑 | |
| └ | functionCallWithValue | Internal 🔒 | 🛑 | |
| └ | functionCallWithValue | Internal 🔒 | 🛑 | |
| └ | _functionCallWithValue | Private 🔐 | 🛑 | |
||||||
| **SafeERC20** | Library |  |||
| └ | safeTransfer | Internal 🔒 | 🛑 | |
| └ | safeTransferFrom | Internal 🔒 | 🛑 | |
| └ | safeApprove | Internal 🔒 | 🛑 | |
| └ | safeIncreaseAllowance | Internal 🔒 | 🛑 | |
| └ | safeDecreaseAllowance | Internal 🔒 | 🛑 | |
| └ | _callOptionalReturn | Private 🔐 | 🛑 | |
||||||
| **Context** | Implementation |  |||
```

└	<Constructor>	Internal 🔒	🛑	
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Internal 🔒	🛑	
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
ERC20	Implementation	Context, IERC20, Ownable		
└	<Constructor>	Public ❗	🛑	NO❗
└	getOwner	External ❗		NO❗
└	name	Public ❗		NO❗
└	decimals	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	transfer	Public ❗	🛑	NO❗
└	allowance	Public ❗		NO❗
└	approve	Public ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	mint	Public ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_burnFrom	Internal 🔒	🛑	
ReentrancyGuard	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
XGrowth	Implementation	Ownable, ReentrancyGuard		
└	<Constructor>	Public ❗	🛑	NO❗
└	_getNextDepositID	Private 🔐		
└	_incrementDepositID	Private 🔐	🛑	
└	pendingReward	Public ❗		NO❗
└	getBalance	Public ❗		NO❗
└	deposit	Public ❗	🛑	nonReentrant
└	claimAllReward	Public ❗	🛑	nonReentrant
└	safeTokenTransfer	Internal 🔒	🛑	
└	setDevFeeAddress	External ❗	🛑	onlyOwner
└	setLendXAddress	External ❗	🛑	onlyOwner
└	setLendXFee	External ❗	🛑	onlyOwner
└	setDevFeeBP	External ❗	🛑	onlyOwner
└	updateApr	External ❗	🛑	onlyOwner
└	withdrawFundsForExternal	External ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, 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.

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

✓ No volatile code.
✓ No high severity issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Saferico hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. 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.