InvestaX

IXAPE Token

SMART CONTRACT AUDIT

29.09.2022

Made in Germany by Chainsulting.de

Table of contents

1. Disclaimer

The audit makes no statements or warrantees about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of IC SG PTE. LTD. If you are not the intended receptor of this document, remember that any disclosure, copying or dissemination of it is forbidden.

Major Versions / Date	Description
0.1 (19.05.2022)	Layout
0.4 (20.05.2022)	Automated Security Testing Manual Security Testing
0.5 (22.05.2022)	Verify Claims and Test Deployment
0.6 (24.05.2022)	Testing SWC Checks
0.9 (24.05.2022)	Summary and Recommendation
1.0 (24.05.2022)	Final document
1.1 (28.06.2022)	Re-check and feedback from client
1.2 (29.09.2022)	Added deployed contract

2. About the Project and Company

Company address:

IC SG PTE. LTD.
100 TRAS STREET, #16-01, 100 AM
Singapore 079027

Website: https://investax.io

Twitter: https://twitter.com/investax

Telegram: https://t.me/investaxtelegram

LinkedIn: https://www.linkedin.com/company/investax

Medium: https://medium.com/@investax

YouTube: https://www.youtube.com/channel/UCAJI2c_gP8TUbaKOrjMX0IA

2.1 Project Overview

Founded in 2015, InvestaX is a MAS licensed digital securities platform offering end-to-end solutions for the issuance, trading and custody of digital securities for private market assets. InvestaX uses blockchain technologies to develop leading technology-driven investment vehicles reducing costs, removing friction, increasing transparency and delivering secondary market trading in private capital markets.

InvestaX launches a new digital asset security tied to an NFT, another groundbreaking investment product that leverages the best of DeFi (Decentralized Finance) and CeFi (Centralized Finance) that changes the way you can interact and give value back to your investor community. Bored Ape #2371, a Bored Ape Yacht Club (BAYC) NFT that belongs to the most sought-after blue-chip collection to date.

BAYC is a technologically distributed, decentralized, community owned asset with its own currency and metaverse. InvestaX tokenized economic interests in #2371 with the ticker symbol - IXAPE. In another world-first project, InvestaX is giving away 1,000 IXAPE tokens to investors and IXAPE will also be tradable on the platform.

The next generation of investment products are using smart contracts and blockchain technologies, creating immense value for those communities involved. What makes BAYC game-changing is that essentially, you have a community-owned asset that also has value on its own, allowing you to create your own BAYC business using the “collective” brand value. IXAPE token holders will get an opportunity to explore an entirely new class of digital assets and participate in the potential appreciation and other financial returns associated with owning these assets.

3. Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. 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 executing the 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

4. Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

4.1 Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
   i. Review of the specifications, sources, and instructions provided to Chainsulting to make sure we understand the size, scope, and functionality of the smart contract.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Chainsulting describe.
2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
   ii. Symbolic execution, which is analysing a program to determine what inputs causes 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.

5. Metrics

The metrics section should give the reader an overview on the size, quality, flows and capabilities of the codebase, without the knowledge to understand the actual code.

5.1 Tested Contract Files

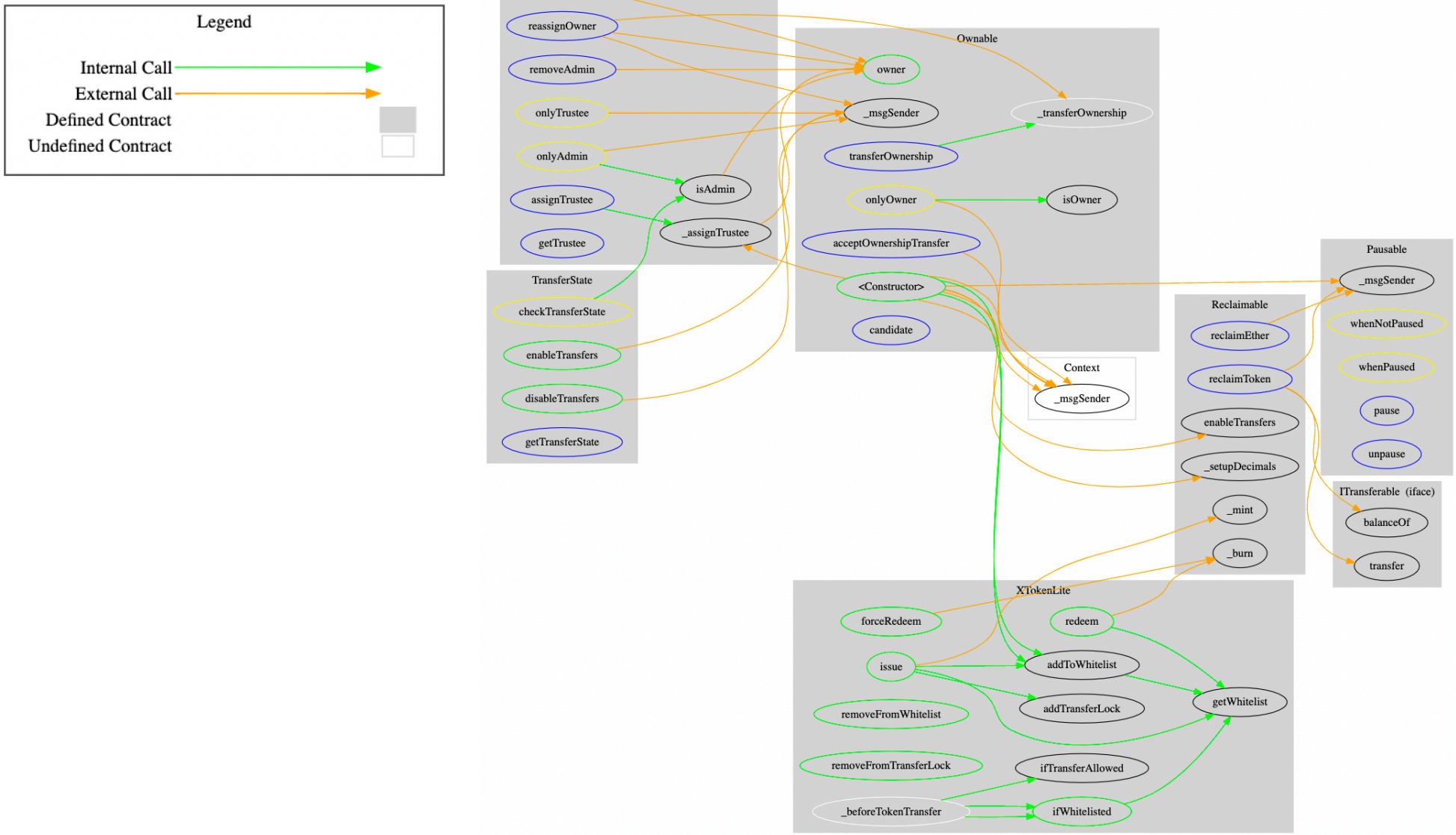
The following are the MD5 hashes of the reviewed files. A file with a different MD5 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different MD5 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

File	Fingerprint (MD5)
./contracts/XToken-Lite.sol	ef64f9698d1d1da46c9d50a15cc61c6a
./contracts/utils/Pausable.sol	0d7904fd5b3660a42954249e65a95943
./contracts/utils/Ownable.sol	4199b587044053e952b49ca9cb3b8445
./contracts/utils/TransferState.sol	571f54913013b34e172cdcf5de335154
./contracts/utils/InvestaAdmin.sol	c01fbd8567803cb45c4836bc75a1b7d0
./contracts/utils/Reclaimable.sol	328972ba0fcd4465567c8a50c8ad3111
./contracts/utils/ITransferable.sol	aa6d184db32148be635376e9536f826f

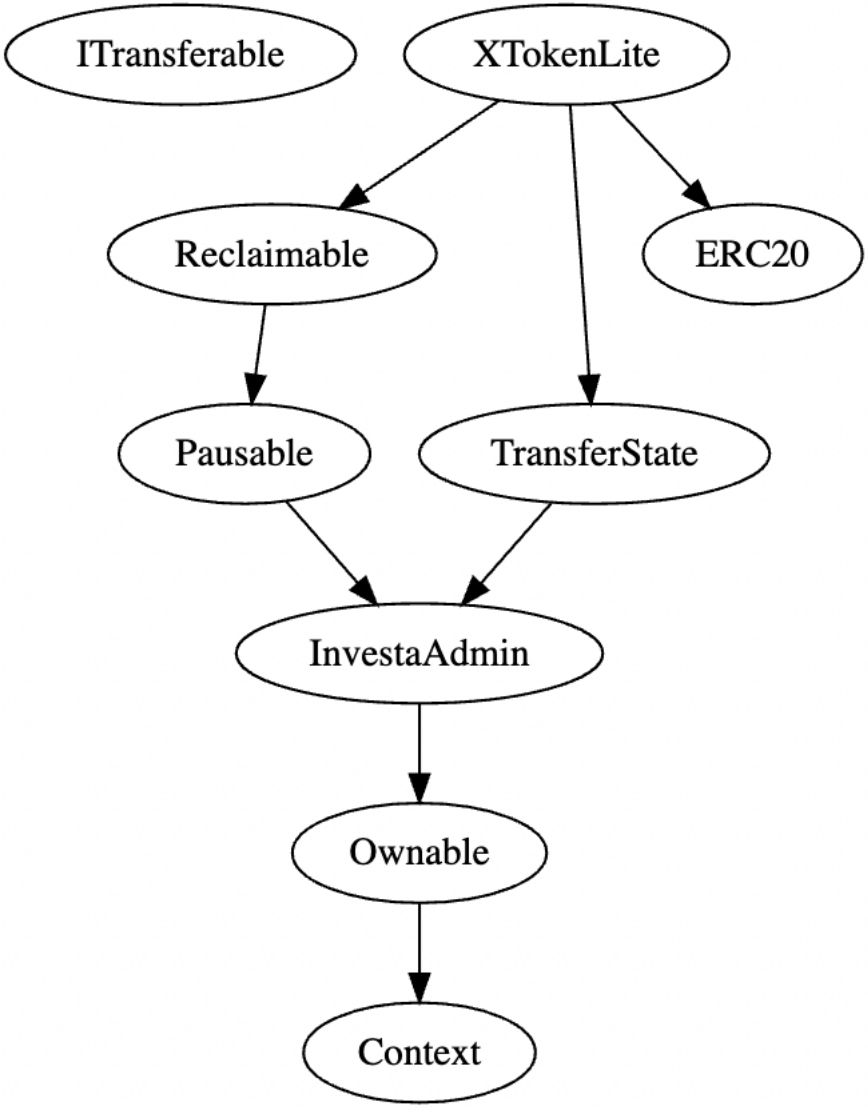
5.2 Used Code from other Frameworks/Smart Contracts (direct imports)

Dependency / Import Path	Source
openzeppelin-solidity/contracts/GSN/Context.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v3.1.0/contracts/GSN/Context.sol
openzeppelin-solidity/contracts/token/ERC20/ERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/tree/v3.1.0/contracts/token/ERC20/ERC20.sol

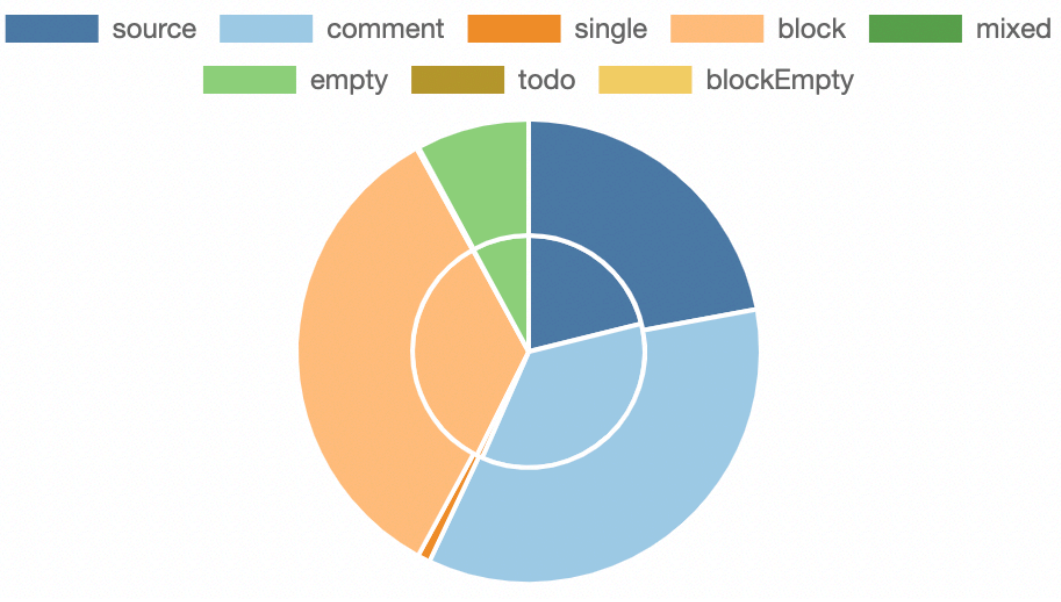
5.3 CallGraph

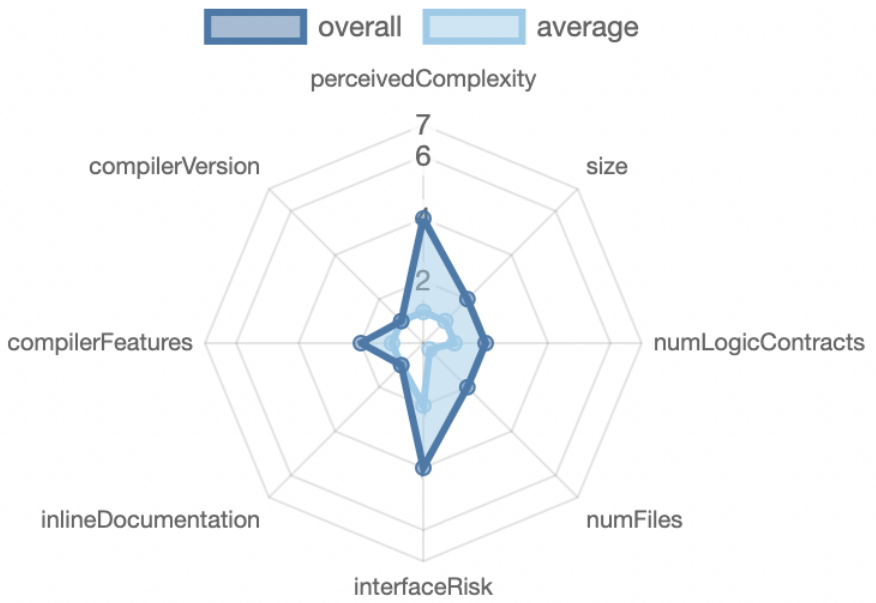


5.4 Inheritance Graph



5.5 Source Lines & Risk





5.6 Capabilities

Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥 Uses Assembly	💣 Has Destroyable Contracts
>=0.4.22 <0.8.0				

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

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
31	0

External	Internal	Private	Pure	View
15	29	0	0	10

StateVariables

Total	🌐Public
8	1

5.7 Source Unites in Scope

Source: https://github.com/InvestaX/x-token-lite
Commit: 94fa5c38e83c90cac9a2037c966a21d9a7f7a5e1

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
🔍	contracts/utils/ITransferable.sol		1	24	21	3	17	5	
🎨	contracts/utils/Reclaimable.sol	1		47	47	15	27	20	🔼
🎨	contracts/utils/InvestaAdmin.sol	1		156	156	60	76	52	
🎨	contracts/utils/TransferState.sol	1		87	87	28	49	22	
🎨	contracts/utils/Ownable.sol	1		148	148	43	87	35	
🎨	contracts/utils/Pausable.sol	1		62	62	23	33	16	
📝	contracts/XToken-Lite.sol	1		300	286	93	156	94	
📝🔍🎨	**Totals**	**6**	**1**	**824**	**807**	**265**	**445**	**244**	🔼

6. Scope of Work

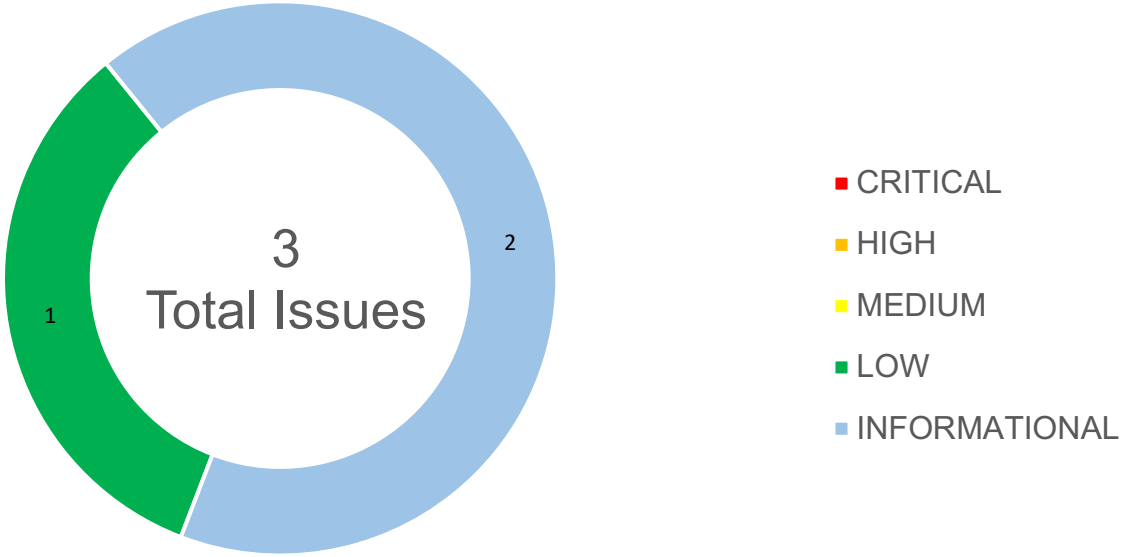
The InvestaX Team provided us with the files that needs to be tested. The scope of the audit is the x-token-lite (IXAPE) contract.

The team put forward the following assumptions regarding the security, usage of the contracts:

- The ERC-20 Token standard is correctly implemented
- Multiple administrators can collectively perform admin-related tasks instead of depending on the owner
- Transfers can only happen between whitelisted addresses
- Transfers can be restricted, and contract can be paused
- Possible to recover accidentally sent ERC-20 compatible tokens and Ethers from the contract
- The smart contract is coded according to the newest standards and in a secure way.

The main goal of this audit was to verify these claims. The auditors can provide additional feedback on the code upon the client's request.

6.1 Findings Overview



No	Title	Severity	Status
6.2.1	Extensive Owner/Trustee Rights	LOW	FIXED
6.2.2	Interface Folder Structure	INFORMATIONAL	ACKNOWLEDGED
6.2.3	Floating Pragma Version Identified	INFORMATIONAL	ACKNOWLEDGED

6.2 Manual and Automated Vulnerability Test

CRITICAL ISSUES
During the audit, Chainsulting‘s experts found **0 Critical issues** in the code of the smart contract.

HIGH ISSUES
During the audit, Chainsulting’s experts found **0 High issues** in the code of the smart contract.

MEDIUM ISSUES
During the audit, Chainsulting’s experts found **0 Medium issues** in the code of the smart contract.

LOW ISSUES
During the audit, Chainsulting’s experts found **1 Low issue** in the code of the smart contract.

6.2.1 Extensive Owner/Trustee Rights
Severity: LOW
Status: FIXED
Code: NA
File(s) affected: InvestaAdmin.sol

Attack / Description	Code Snippet	Result/Recommendation
The trustee has extensive rights, which effects the owner role. The auditor has recognized kind of role	Line: 57 – 81 (`/**` `* @dev Assigns or changes the trustee wallet.`	The owner has extensive rights which can be reassigned by a trustee, those has at the end the most power. If the trustee wallet/private key gets into the wrong hands caused by a leak or hack, then it’s

structure but not any multi-sig structure.	``` * @param account Enter a wallet address which will become the new trustee. * @return Returns true if the operation was successful. * @notice This feature is restricted for owner use only. */ function assignTrustee(address account) external onlyOwner returns (bool) { return _assignTrustee(account); } /** * @dev Internal function to assign or change the trustee wallet. * @param account Enter a wallet address which will become the new trustee. * @return Returns true if the operation was successful. * @notice This feature is restricted for owner use only. */ function _assignTrustee(address account) internal returns (bool) { require(account != address(0), "Invalid address"); require(account != super.owner(), "The owner cannot become the trustee!"); _trustee = account; emit TrusteeAssigned(account); return true; ```	easily possible to assign a new owner and within all owner rights. We recommend protecting the trustee wallet with a multi-signature structure such as gnosis safe 2 signatures of 3 for example, to change the owner.

	}	

INFORMATIONAL ISSUES

During the audit, Chainsulting's experts found **2 Informational issues** in the code of the smart contract.

6.2.2 Interface Folder Structure

Severity: INFORMATIONAL
Status: OPEN
Code: NA
File(s) affected: /utils/ITransferable.sol

Attack / Description	Code Snippet	Result/Recommendation
The project structure should contain an extra folder for interfaces.	NA	It is recommended to move the interface into a separate folder called interfaces. ./interfaces/ITransferable.sol

6.2.3 Floating Pragma Version Identified

Severity: LOW
Status: ACKNOWLEGED
Code: SWC-103
File(s) affected: ALL

Attack / Description	Code Snippet	Result/Recommendation
The current pragma uses semantic versioning operators ">=0.4.22 <0.8.0" which allows multiple compilers to be used.	`pragma solidity >=0.4.22 <0.8.0;`	It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code. i.e. Pragma solidity 0.6.0

6.3 SWC Attacks

ID	Title	Relationships	Test Result
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	✅
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	✅
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	✅
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	✅
SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	✅

ID	Title	Relationships	Test Result
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	✅
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	✅
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	✅
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	✅
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	✅
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	✅
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	✅
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	✅
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	✅

ID	Title	Relationships	Test Result
SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	✅
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	✅
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	✅
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	✅
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	✅
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	✅
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	✅
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	✅
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	✅
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	✅

ID	Title	Relationships	Test Result
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	✅
SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	✅
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	✅
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	X
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	✅
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	✅
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	✅

6.4 Verify Claims

6.4.1 The ERC-20 Token standard is correctly implemented
Status: tested and verified ✅

6.4.2 Multiple administrators can collectively perform admin-related tasks instead of depending on the owner
Status: tested and verified ✅

6.4.3 Transfers can only happen between whitelisted addresses
Status: tested and verified ✅

6.4.4 Transfers can be restricted, and contract can be paused
Status: tested and verified ✅

6.4.5 Possible to recover accidentally sent ERC-20 compatible tokens and Ethers from the contract
Status: tested and verified ✅

6.4.6 The smart contract is coded according to the newest standards and in a secure way
Status: tested and verified ✅

7. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase.

The main goal of the audit was to verify the claims regarding the security and functions of the smart contract. During the audit, no critical, no high, no medium, 1 low and 2 informational issues have been found, after the manual and automated security testing. No necessary need for action, as the recommendations only further enhance the code's readability, not security. The NatSpec documentation within the codebase have greatly benefiting the understanding of the usage and showed us a high professionalism of the InvestaX Team.

Update (28.06.2022): Our auditors have reviewed and approved the multi-signature flow

8. Deployed Smart Contract

VERIFIED

https://polygonscan.com/address/0x9a44041ea059103e1b56c89ed50481f7d84fe700#code

9. About the Auditor

Chainsulting is a professional software development firm, founded in 2017 and based in Germany. They show ways, opportunities, risks and offer comprehensive web3 solutions. Their services include web3 development, security and consulting.

Chainsulting conducts code audits on market-leading blockchains such as Solana, Tezos, Ethereum, Binance Smart Chain, and Polygon to mitigate risk and instil trust and transparency into the vibrant crypto community. They have also reviewed and secure the smart contracts of 1Inch, POA Network, Unicrypt, LUKSO among numerous other top DeFi projects.

Chainsulting currently secures $100 billion in user funds locked in multiple DeFi protocols. The team behind the leading audit firm relies on their robust technical know-how in the web3 sector to deliver top-notch smart contract audit solutions, tailored to the clients' evolving business needs.

Check our website for further information: https://chainsulting.de