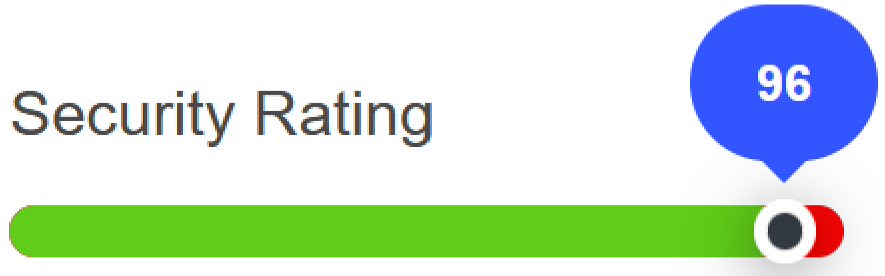
SECURICHAIN

SMART CONTRACT SECURITY ANALYSIS REPORT ON

EPIC-WARS

Feb 28th 2022



(The rating is based on the number, severity and latest status of detected issues)

Disclaimer

This report containings confidential information which can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities are fixed — upon a decision of the Customer.

SecuriChain does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed.

The report in no way provide investment advice, nor should be leveraged as investment advice of any sort.

TABLE OF CONTENTS

1. VULNERABILITY ASSESSMENT OVERVIEW

1.1. ASSIGNING RISK LEVELS

The Auditor categorizes each of the detected vulnerabilities into 4 levels（**High**, **Medium**, **Low**, and **Info**) according to the degree of the risks it may cause in Customer's operations. For details of the rating standards, please refer to "Appendix 2 Risk Rating." Please also note that the assessment of the findings is based on Auditor's own perspective and may contain speculations in some cases.

1.2. SCOPE OF WORK

Project Name	EPICWAR
Platform	ETHEREUM
Languages	SOLIDITY
Methods	AUTOMATION SCAN, ARCHITECTURE REVIEW, FUNCTIONAL TESTING, MANUAL CODE REVIEW
Repository	EPICWAR-MARKETPLACE-CONTRACTS COMMIT: 264B284 MYSTERY-BOX-CONTRACT COMMIT: 5C39A40
Documents	
Timelines	*Feb 14th 2022 - Feb 28th 2022*

1.3. CHECKSUM FILE

EPICWAR-MARKETPLACE-CONTRACTS

STT	Hash	Name
1	84527c0bf2f7ced36c6db14d327d28cfa5c4922037652f305c52b5aabe077d93	MarketplaceAuction.sol
2	531946a7d2df2dbda9771474e007d1bc3bcae415593520a03121f777bd74df54	Marketplace.sol
3	9efc8eb762c3e7ad0cfa6cfdd7b9307d505681e9263d0801828ff67f5a594f46	TokenTest.sol
4	9107461b53e25c5e25e166440722c3a2740df906243a715abc3c1cfe3a970d71	WETH.sol
5	843460b7be1bac92658f43dc5df80d43ab36f62f5881016f367106e25b4706b8	IMarketplace.sol
6	55267256fc784ccf4fbf4d50347f78f40a1bb67be95dce281f1dd09547fac294	IWETH.sol

MYSTERY-BOX-CONTRACT

STT	Hash	Name
1	cad2aae2d25a18cd51e010b6f99dee7933de1ced3bea720283b832e7fa579e69	EpicWarBox.sol
2	4841461918a9a1b046ea4ed60391099ef15a2065ee162ad97d23ab03998ef87a	EpicWarNFT.sol
3	dc019c0699079086f8242dfa31c40239b0fbe05592f993b513c00cbedcad41b2	EpicWarNumber.sol
4	9efc8eb762c3e7ad0cfa6cfdd7b9307d505681e9263d0801828ff67f5a594f46	TokenTest.sol

1.4. ASSESSMENT RESULTS

According to the assessment, the Customer's smart contracts have the security rating of 96/100

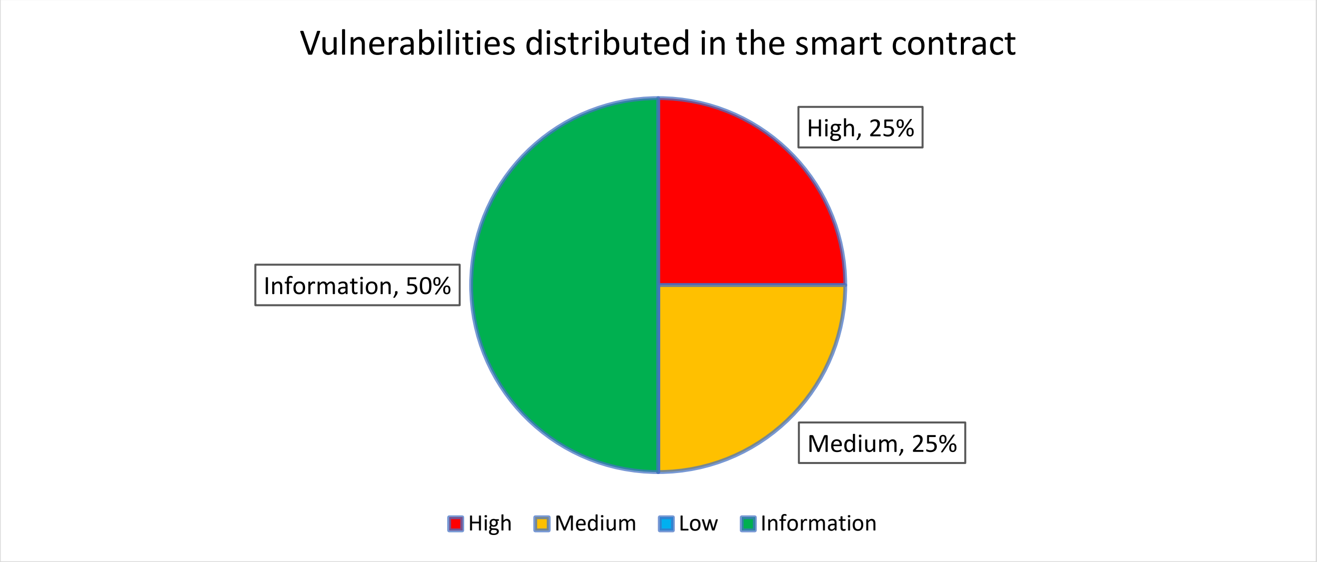
Rate	Description
96-100	**No vulnerabilities** were found or all detected ones have been resolved
70-95	Unresolved **Low-level** vulnerabilities exist
40-69	Unresolved **Medium-level** vulnerabilities exist
0-39	Unresolved **High-level** vulnerabilities exist

（For more information on criteria for risk rating, refer to Appendix.2)

2. FINDINGS

2.2. LIST OF VULNERABILITIES

The detected vulnerabilities are listed below. Please refer to "Appendix.2 Risk Rating" for the risk assessment method.



ID	Risk Level	Name	Amount	Status
SC1	Information	Unlocked Pragma	2	Unresolved
SC2	Medium	Logic vulnerability	1	Resolved in #1aa00ae commit
SC3	High	DoS vulnerability	1	Resolved in #1aa00ae commit

(For rating of each vulnerability, refer to Appendix 2.)

2.3. DETAILS

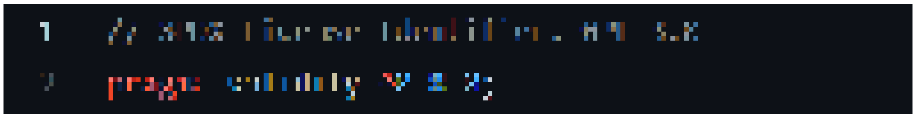
[1] Unlocked Pragma

2 INFO

- **Overview**

Contracts should be deployed with the same compiler version and flags that they have been thoroughly tested. Locking the pragma helps to ensure that contracts do not accidentally get deployed using.

- **Possible Impact**



(Blurring the image of the code snippet in the public report because the Customer's code is in the private repository)

An outdated compiler version that might introduce bugs that affect the contract system negatively.

- **Recommendation**

Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the chosen compiler version.

Pragma statements can be allowed to float when a contract is intended for consumption by other developers, as in the case with contracts in a library or EthPM package. Otherwise, the developer would need to manually update the pragma in order to compile locally.

- **Location:**
  - EPICWAR-MARKETPLACE-CONTRACTS:: ALL CONTRACT
  - MYSTERY-BOX-CONTRACT:: ALL CONTRACT

[2] Logic vulnerability in takeOffer() function

1 MEDIUM

- **Overview**

Using the takeOffer() function can delete (unlist) any product on the market.

- **Possible Impact**

(Blurring the image of the code snippet in the public report because the Customer's code is in the private repository)

Perform ‘marketitemid’ deletion without checking its owner.

- **Recommendation**

Check the owner of ‘marketitemid’ before removing it.

- **Location:**
    - EPICWAR-MARKETPLACE-CONTRACTS: (L153-L177)

1 HIGH

[3] DoS with (unexpected) revert

- **Overview**

The attacker can win the auction with the smallest price.

- **Possible Impact**

(Blurring the image of the code snippet in the public report because the Customer's code is in the private repository)

1. The attacker first writes a contract to bid on.
2. When someone bids higher, the Contract will return the money to attacker
3. When the funds are returned, the attacker's fallback() function will call revert() causing the transaction to fail
4. Since the transaction that returned the funds to the attacker was faulty, other users can not bid higher.

- **Recommendation**

In view of the above situation, if the result of the external function call needs to be processed before entering the new state, it must be considered that the external call might fail anytime.

- **Location:**
  - EPICWAR-MARKETPLACE-CONTRACTS: (L106-L123)

3. CONCLUSION

This document, and its appendices, represents the results of several days of our intensive work.

Smart contracts within the scope were analyzed with static analysis tools and manually reviewed.

Please feel free to direct any questions on this assessment to: audit@securichain.io.

APPENDIX 1. ASSESSMENT LIST

CHECKLIST		
Arithmetic operations		
	Integer Overflow/Underflow	Integer Division
	Integer Truncation	Integer Sign
	Wrong Operator	
Re-entrancy		
Bad Randomness		
	Timestamp Dependence	Blockhash
Front running		
DDos		
	DOS By Complex Fallback Function	DOS By Gaslimit
	DOS By Non-existent Address Or Malicious Contract	
Unsafe external calls		
Gas usage		
	Invariants in Loop	Invariants State Variables Are Not Declared Constant
Business Logics Review		
Access Control & Authorization		
	Replay Attack	Use tx.origin For Authentication
Logic Vulnerability		

APPENDIX 2. RISK RATING

Risk Level	Explain	Example Types
High	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.	Re-entrancy Front running DDos Bad Randomness Logic Vulnerability Arithmetic operations
Medium	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.	Access Control Unsafe external calls Business Logics Review Logic Vulnerability
Low	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.	Gas usage
Info	The issue does not pose an immediate risk, but is relevant to security best practices or Defence in Depth.	Do not specify a specific version of Solidity