Explorando e classificando bugs comumente encontrados em contratos inteligentes

Ana Julia

October 1, 2023

Contents

references

1 Introdução

A tecnologia blockchain, primeiramente introduzida por Satoshi Nakamoto em 2008, é identificada como uma megatendência computacional capaz de revolucionar múltiplos setores industriais[**1**]. As características distintas de segurança, transparência e rastreabilidade inerentes à blockchain têm incentivado uma ampla gama de setores a explorar seu uso na reestruturação de suas operações fundamentais. A aplicabilidade dessa tecnologia ultrapassa o domínio das criptomoedas, abarcando setores como pagamentos, gerenciamento de identidade, saúde, eleições governamentais e outros[**2**].

A publicação do whitepaper do Ethereum em 2014 simbolizou um avanço considerável na evolução da tecnologia blockchain[**3**]. Diferentemente do Bitcoin, concebido originalmente como uma moeda digital, o Ethereum inaugurou uma funcionalidade disruptiva no campo da tecnologia blockchain: os contratos inteligentes. A inovação trazida pelo Ethereum reside na incorporação de uma máquina virtual capaz de processar códigos em linguagens de programação *Turing complete* na blockchain, habilitando assim a construção de aplicativos descentralizados. Estes aplicativos propõem a substituição dos sistemas de back-end por contratos inteligentes que operam em uma blockchain[**7**]. Devido as características inerentes a tecnologia blockchain, como a transparência e descentralização, os aplicativos que rodam no Ethereum são sucetíveis a vulnerabilidades que podem ser exploradas por hackers, resultando em grande prejuízo financeiro para os

protocolos e utilizadores dos mesmos. Apenas no primeiro trimestre de 2023, 320 milhões de dólares foram perdidos devido a ataque de hackers no Ethereum[**1**].

[**1**]

2 Revisão Bibiográfica

O que é EVM, EOA, contracts, transactions (nonce).

3 Metodologia

3.1 Perguntas

- Categorizando bugs

3.2 Categorias dos protocolos

- Liquid Staking: Protocols that enable you to earn staking rewards on your tokens while also providing a tradeable and liquid receipt for your staked position
- Lending: Protocols that allow users to borrow and lend assets
- Dexes: Protocols where you can swap/trade cryptocurrency
- Bridge: Protocols that bridge tokens from one network to another
- CDP: Protocols that mint its own stablecoin using collateralized lending
- Services: Protocols that provide a service to the user
- Yield: Protocols that pay you a reward for your staking/LP on their platform
- RWA: Protocols that involve Real World Assets, such as house tokenization
- Derivatives: Protocols for betting with leverage
- Yield Aggregator: Protocols that aggregated yield from diverse protocols

- Cross Chain: Protocols that add interoperability between different blockchains
- Synthetics: Protocol that created a tokenized derivative that mimics the value of another asset.
- Launchpad: Protocols that launch new projects and coins
- Indexes: Protocols that have a way to track/created the performance of a group of related assets
- Liquidity manager: Protocols that manage Liquidity Positions in concentrated liquidity AMMs
- Insurance: Protocols that are designed to provide monetary protections
- Privacy: Protocols that have the intention of hiding information about transactions
- Infrastructure
- Algo-Stables: Protocols that provide algorithmic coins to stablecoins
- Payments: Protocols that offer the ability to pay/send/receive cryptocurrency
- Leveraged Farming: Protocols that allow you to leverage yield farm with borrowed money
- Staking Pool: Refers to platforms where users stake their assets on native blockchains to help secure the network and earn rewards. Unlike Liquid Staking, users don't receive a token representing their staked assets, and their funds are locked up during the staking period, limiting participation in other DeFi activities
- NFT Marketplace: Protocols where users can buy/sell/rent NFTs
- NFT Lending: Protocols that allow you to collateralize your NFT for a loan
- Options: Protocols that give you the right to buy an asset at a fixed price
- Options Vault: Protocols that allow you to deposit collateral into an options strategy

- Prediction Market: Protocols that allow you to wager/bet/buy in future results
- Decentralized Stablecoin: Coins pegged to USD through decentralized mechanisms
- Farm: Protocols that allow users to lock money in exchange for a protocol token
- Uncollateralized Lending:Protocol that allows you to lend against known parties that can borrow without collaterall
- Reserve Currency: OHM forks: Protocols that uses a reserve of valuable assets acquired through bonding and staking to issue and back its native token
- RWA Lending: Protocols that bridge traditional finance and blockchain ecosystems by tokenizing real-world assets for use as collateral or credit assessment, enabling decentralized lending and borrowing opportunities.
- Gaming: Protocols that have gaming components
- Oracle: Protocols that connect data from the outside world (off-chain) with the blockchain world (on-chain)
- P2P File distributoin system
- DAO

Fonte: `https://defillama.com/categories`

3.3 Classificação dos bugs

- O2: We cannot access the source code of the project.
- O4: Bugs that occur in off-chain components
- C3: Erroneous state updates.
  - S3-1: Missing state update.
  - S3-2: Incorrect state updates, e.g., a state update that should not be there.
- C5: Privilege escalation and access control issues.

  - C5-1: Users can update privileged state variables arbitrarily (caused by lack of ID-unrelated input sanitization).
  - C5-2: Users can invoke some functions at a time they should not be able to do so.
  - C5-3: Privileged functions can be called by anyone or at any time.
  - C5-4: Funds can get locked due to missing withdraw code

- C6: Erroneous accounting.
  - C6-1: Incorrect calculating order.
  - C6-2: Returning an unexpected value that deviates from the expected semantics specified for the contract.
  - C6-3: Calculations performed with incorrect numbers (e.g., x = a + b ==> x = a + c).
  - C6-4: Other accounting errors (e.g., x = a + b ==> x = a - b).

- C7: Broken business logic
  - C7-1: Unexpected function invocation sequences (e.g., external calls to dependent contracts).
  - C7-2: Unexpected environment or contract conditions (e.g., ChainLink returning outdated data or significant slippage occurring).
  - C7-3: A given function is invoked multiple times unexpectedly.
  - C7-4: Unexpected function arguments.

- C8: Contract implementation-specific bugs. These bugs are difficult to categorize into the above categories.

- C9: Lack of signature replay protection, e.g missing nonce

- C10: Missing check. Missing Check refers to a critical oversight in a smart contract's code where a necessary condition or validation is not properly implemented.

3.4 Dados coletados

Foi feito a curadoria de 477 bugs classificados com severidade alta

Plataforma	Protocolo	Categoria do protocolo	N de auditores	Descrição
Sherlock	Perennial V2	Derivatives	4	Oracle request time
Sherlock	Perennial V2	Derivatives	1	Invalid oracle versio
Sherlock	Perennial V2	Derivatives	4	Protocol fee from M
Sherlock	Perennial V2	Derivatives	3	PythOracle:if price
Sherlock	Perennial V2	Derivatives	4	Vault.sol: settleing
Sherlock	Perennial V2	Derivatives	1	Keepers will suffer
Sherlock	Blueberry	Leverage Farming	1	Stable BPT valuati
Sherlock	Blueberry	Leverage Farming	2	CurveTricryptoOra
Sherlock	Blueberry	Leverage Farming	2	CurveTricryptoOra
Sherlock	Blueberry	Leverage Farming	1	CVX/AURA distri
Sherlock	Blueberry	Leverage Farming	1	wrong bToken's exc
Code4Arena	Arbitrum Foundation	DAO	3	Signatures can be r
Code4Arena	PoolTogether	Yield	1	Too many rewards
Code4Arena	PoolTogether	Yield	16	rngComplete functi
Sherlock	Tokensoft	Launchpad	24	"Votes" balance ca
Sherlock	Bond Options	Options	14	All fund from Telle
Sherlock	Bond Options	Options	4	All funds can be st
Sherlock	Symmetrical Update	Derivatives	2	liquidatePartyA rec
Sherlock	Symmetrical Update	Derivatives	2	liquidatePositionsP
Sherlock	Cooler Update	Lending	3	Can steal gOhm by

4 Desenvolvimento #encontrar nome melhor

4.1 Categorias

4.2 Dificuldade