Poster: Demystifying Exploitable Bugs in Smart Contracts

Zhuo Zhang*, Brian Zhang†, Wen Xu‡§, Zhiqiang Lin‖
*Purdue University, †Harrison High School, ‡Georgia Institute of Technology, §PNM Labs, ‖Ohio State University,
zhan3299@purdue.edu, bzhangprogramming@gmail.com, wen@pnm.xyz, zlin@cse.ohio-state.edu

I. Publication Information

Our research will be published in the proceedings of the International Conference on Software Engineering (ICSE) 2023 [1]. Several media outlets have also covered our findings, including *Week in Ethereum News* [2] and an engaging interview with *Code4rena* [3]. We have made our dataset openly available to encourage further exploration and academic contributions in this field. Details can be found in our GitHub repository [4].

II. Extended Abstract

Since the Bitcoin and blockchain technology were introduced in 2008, their market capitalization has experienced an explosive growth, reaching over $438 billion (as of 5 August 2022) [5]. Nowadays, there exists countless blockchain-based products and services for anyone to interact with, such as those in travel, healthcare, finances, and lately virtual reality. Blockchains such as Ethereum, Solana, and Polygon handle millions of transactions everyday. High-level programming languages like Solidity enable the creation and integration of numerous innovative ideas with blockchains, in the form of *smart contracts*. Just like traditional software applications, smart contracts are composed by developers and hence susceptible to human errors. Many of them are exploitable. According to [6], $1.57 billion were exploited from various smart contracts as of 1 May 2022.

A large body of techniques have been proposed to detect smart contract vulnerabilities such as reentrancy and integer overflows, and they can be classified into categories such as fuzzing, formal verification, and runtime verification. Despite the success of these techniques, smart contract exploits are still commonly seen in the wild [7]. This may root at the fundamental differences between smart contract and traditional software vulnerabilities.

Differences between Smart Contract and Traditional Software Vulnerabilities. For traditional software, security vulnerabilities are largely different from functional bugs. The former has limited forms such as buffer overflow (leading to control flow hijacking), information leak, and privilege escalation, whereas the latter is very diverse, denoting violations of domain-specific and even application-specific properties. Moreover, functional bugs in traditional software usually lead to incorrect outputs and/or interrupted services, which may not cause direct security concerns. In contrast, smart contract vulnerabilities are in many cases functional bugs, due to their unique nature, incorrect outputs in smart contract usually indicate monetary loss. Finding these vulnerabilities hence requires checking domain-specific properties, which is much harder than checking a limited set of general security properties in traditional software.

Therefore, we consider that it is highly valuable to summarize recent exploitable smart contract bugs to understand the underlying critical properties. In this research, we study a large set of 516 exploitable bugs from 167 real-world contracts reported/exploited in years 2021-2022, and aim to summarize their root causes and distributions. We collect these bugs from the highly reputable *Code4rena* contests [8] (with a total of 462 bugs), which invite individuals and companies from all over the world to audit real-world contracts by providing substantial bounties [8], and from various real-world exploit reports, with a total of 54 exploits. The real-world exploits account for $256.3 millions monetary loss. In the study, we answer a few research questions such as how many such bugs can be detected by existing tools, how difficult is it to detect such bugs, the root causes of those that cannot be detected by tools, their consequences, repair strategies, and distributions. The detailed setup of our study is in §III. Our findginds are highlighted in the following.

- Although the DeFi community has heavily invested on protecting their products, the current supply of tools and human auditor resources have not met the demand.
- Existing techniques rely on simple and general oracles or hand-coded ones that are project specific. Such oracles may not be sufficient for functional bugs in general.
- More than 80% exploitable bugs are beyond existing tools (we call them *machine unauditable bugs* (MUBs)). This is largely due to the lack of describing and checking the corresponding domain-specific properties.
- Majority of exploitable bugs in the wild are hard to find, including those within and beyond the scope of tools.
- MUBs can be classified to seven categories. Two of the categories (accounting for 40% of the MUBs) are project/implementation specific (consequently no general oracles to detect them). The remaining five categories have clear symptoms and can be properly abstracted such that automated oracles may be devised.
- Different types of MUBs have different distributions and different difficulty levels, with *price oracle manipulation* (38%) and *privilege escalation* most popular in real-world

exploits, and *accounting errors* most popular in bugs found during audit contests.
- Different kinds of DeFi projects tend to be prone to different types of MUBs.

We demonstrate the importance of our findings by our preliminary success in finding 15 zero-day exploitable bugs in real-world smart contracts. These bugs could endanger $22.52 millions funds if exploited. We have been rewarded with $102,660 bug bounties for identifying these bugs.

III. Dataset Information

We collect two datasets of bugs, from the Code4rena contests and real-world exploit reports.

Code4rena Contests. Code4rena [8] is a highly reputable audit contest platform. Each Code4rena contest lasts for 3-7 days and aims to have real-world DeFi projects audited *before official deployment* (pre-deployment), for which the developers of subject projects commit a bounty in the range of $20K to $1M as incentive. Individuals, companies, and institutes from all over the world can participate. After the contest, a group of Code4rena judges (i.e., very experienced auditors elected by the community) and the project's developers get together to inspect the bug reports, where they confirm the valid ones, classify reports based on root causes, and decide the criticality level of bugs. Note that each bug is assigned a criticality level: low, medium, or high, where only high-risk bugs can cause asset loss (and hence are exploitable) [9]. The final reward is decided by both the criticality level of the bug and the number of reports submitted for the bug (more submissions lead to a lower reward as the bug is easier than others).

We collect and analyze 462 unique high-risk bugs from 113 Code4rena contests hosted between April 2021 and June 2022. For each case, we inspect the bug report, the faulty contracts (which are available through Github), and the project's documentation. Following the suggestions in Claes et al. [10], each bug is checked by at least two individual researchers. Any disagreement will be turned to an additional researcher. We reach consensus for all cases after the new researcher gets involved. All our researchers are experienced auditors, having participated 23 contests from February 2022 to June 2022. One of them was invited to be a consultant for judges.

Among the 462 surveyed bugs, we identify 341 in-scope bugs (exploitable by remote users). Table I presents the basic information of surveyed contests and the in-scope bugs. The first column presents the categories of on-chain projects, following the taxonomy by DefiLlama [11], a leading DeFi analytics platform. The description of each category is available in our supplementary material [4] (§II). Observe that around $2.8 billions are protected by Code4rena auditing, indicating the representativeness of the dataset, and $6.7 millions are committed as bounties.

Real-world Exploits. Our second dataset comprises 54 real-world exploits, collected from postmortems and bugfix reviews of real-world exploits from January 2022 to June 2022. These reports are published by highly-reputable security researchers (e.g., [12], [13]) and companies (e.g., [14]–[17]). We follow the aforementioned study methodology (for Code4rena reports). Overall, we identify 44 (out of 54) in-scope bugs. Specifically, real-world exploits target *post-deployment* contracts, including real attacks launched against on-chain contracts and caused real asset damage (i.e., *attacks*), and the cases in which ethical hackers demonstrated vulnerabilities in a local off-chain environment and were awarded bug bounties by the projects (i.e., *bug bounties*). over $265 million were lost due to real attacks in the first half of 2022; despite the substantial auditing efforts paid prior to deployment, there are still many post-deployment exploitable bugs.

TABLE I: Basic information of Code4rena contests. **# Cont** and **# Vuln** denote the numbers of hosted contests and in-scope bugs, respectively. **# Atten** denotes the number of auditors who have attended at least one contest of the corresponding category, while the total **# Atten** denotes the total number of auditors who have ever participated in Code4rena contests. **TVL** denotes the overall value of crypto assets deposited in the corresponding DeFi projects, i.e., the worth of these projects.

Categories	# Cont	Bounty	# Atten	# Vuln	TVL
Lending	20	$1,145K	180	53	$304.8M
Dexes	13	$1,020K	139	70	$898.9M
Yield	12	$ 970K	193	85	$304.8M
Services	11	$ 532K	123	21	$219.8M
Derivatives	9	$ 525K	123	13	$147.8M
Yield Aggregator	9	$ 365K	124	22	$265.5M
Real World Assets	7	$ 405K	69	10	$ 41.8M
Stablecoins	6	$ 365K	102	7	$364.7M
Indexes	6	$ 215K	101	7	$ 1.0M
Insurance	5	$ 298K	74	19	$ 42.9M
NFT Marketplace	4	$ 266K	126	8	$ 46.6M
NFT Lending	4	$ 230K	108	10	$ 8.2M
Cross Chain	4	$ 250K	81	7	$ 32.0M
Others	3	$ 110K	25	9	$118.3M
Total	113	$6.696M	358	341	$2.797B

References

[1] Z. Zhang, B. Zhang, W. Xu, and Z. Lin, "Demystifying exploitable bugs in smart contracts," *ICSE*, 2023.
[2] "Week in ethereum news march 4, 2023." [Online]. Available: https://weekinethereumnews.com/week-in-ethereum-news-march-4-2023/
[3] "Demystifying exploitable bugs in smart contracts with zhuo and brian." [Online]. Available: https://shorturl.at/itY04
[4] "Web3bugs." [Online]. Available: https://github.com/ZhangZhuoSJTU/Web3Bugs
[5] "Bitcoin market cap." [Online]. Available: https://coinmarketcap.com/
[6] "The growing rate of defi fund loss." [Online]. Available: https://twitter.com/PeckShieldAlert/status/1520620826613010432
[7] "The nine largest crypto hacks in 2022." [Online]. Available: https://blockworks.co/the-nine-largest-crypto-hacks-in-2022/
[8] "Code4rena." [Online]. Available: https://code4rena.com
[9] "Judging criteria - code4rena." [Online]. Available: https://docs.code4rena.com/awarding/judging-criteria#estimating-risk
[10] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.
[11] "Defillama." [Online]. Available: https://defillama.com/
[12] "samczsun." [Online]. Available: https://twitter.com/samczsun
[13] "pwning.eth." [Online]. Available: https://twitter.com/PwningEth
[14] "Peckshield." [Online]. Available: https://twitter.com/peckshield
[15] "Paradigm." [Online]. Available: https://twitter.com/paradigm
[16] "Certik." [Online]. Available: https://twitter.com/CertiK
[17] "Immunefi." [Online]. Available: https://immunefi.com/explore/