Decentralized Finance Withdrawal Delays

Jan Gorzny

Zircuit

Introduction

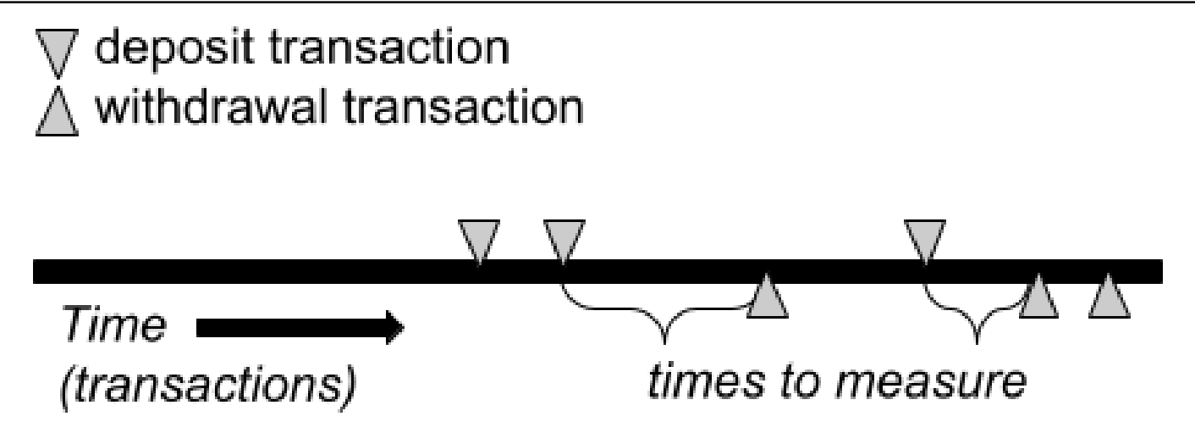
Decentralized Finance ("DeFi"; see e.g., [2]) is a major use case of general purpose blockchains like Ethereum. DeFi refers to financial services implemented as *decentralized Applications* ("dApps") on these blockchains. They use on-chain assets for a myriad of purposes, including market making, loan issuance, and stablecoins. These use cases mean that DeFi protocols handle billions of dollars of cryptocurrency, which in turn makes them an attractive target for exploitation [5].

In [1], the authors argue that preventing withdrawals after deposits in DeFi protocols may have safeguarded millions of dollars in cryptocurrency. Enforcing a suitable delay of more than one block or a non-zero amount of time (according to the block timestamps), can outright prevent so-called flash loan attacks (see e.g. [4]) by making such a loan too expensive. Despite this benefit, this delay may be undesirable. One reason for this is that it may break *composability*, a defining characteristic of DeFi protocols [3]: the ability to build DeFi protocols on top of other ones. Other reasons may include the introduction of a more restrictive user experience (especially with a poorly chosen delay) and the increase of on-chain gas costs.

Contributions. We analyse public blockchain data to determine the minimum, maximum, and average duration (in blocks) between a call to a deposit function and a corresponding withdrawal function for various DeFi protocols on Ethereum. This results in a first step towards understanding if delays really break composability or should be considered for additional security. We show that for some well-used protocols, a mandatory delay between deposits and withdrawals would not negatively affect user experience.

Measured Values

The values we want to measure are illustrated in the figure to the right.



Let X_p be the set of addresses on Ethereum that call a withdrawal or deposit function on a contract ρ in the block range $[MIN, MAX]$. We also count the total number of transactions calling either a deposit function d_ρ, withdrawal w_ρ, and the total t_ρ. Let $\delta_\rho^{(i,x)}$ be the ith delay between a deposit and subsequent withdrawal for a protocol ρ for an address (user) x counted in blocks. We want δ_ρ: the minimum observed delay between deposits and withdrawals for a protocol ρ, which is $\delta_\rho = \min_{x,i} \delta_\rho^{(i,x)}$. We also collect the maximum observed delay (Δ_ρ) as well as the average a_ρ delay in blocks.

Dataset

We chose three DeFi protocols to analyse: Curve, Sturdy, and Yearn.

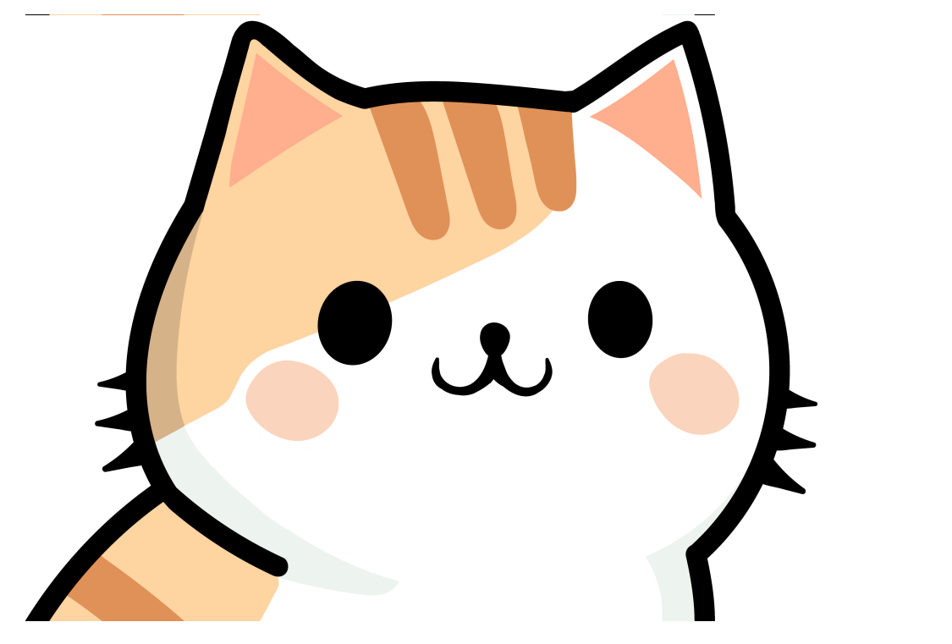
- Curve Vyper contracts; chose the one highlighted in post-mortem.
- Sturdy Solidity contracts; chose the main proxy contract highlighted in a post-mortem where 513 WETH was borrowed.
- Yearn Vyper contracts; chose the proxy contracts of the DAI and WETH V2 pools. These were the top two V2 Yearn lending pools with the most Total Value Locked (TVL) at the time of writing ($17M and $61M USD respectively)

These protocols were exploited in 2023 and were the only protocols in [1] on Ethereum mainnet. Data was collected from block $MIN = 16308190$ (Jan-01-2023 12:00:11 AM +UTC) to $MAX = 17595510$ (Jul-01-2023 12:00:11 AM +UTC); after contracts were deployed but before they were exploited. This represents the first six months of blocks (1287320 blocks) on Ethereum in 2023 We identified functions on each contract that allow users to deposit and withdraw functions. These functions and the contract addresses are in Table 1.

Discussion

Table 1 displays the data collected. Observe that most contracts have a relatively small amount of deposits and withdrawals, suggesting that most interactions actually happen through indirect calls. The average deposit values a_p are rounded to the nearest integer but are always large: the average deposit for most users is at least $432,472$ blocks for all protocols (≈ 60 days). A key finding is that the smallest gap is 4 blocks: meaning that any restriction to prevent a flash loan within a single transaction will not affect users who interact with these protocols via these function calls.

Results



Protocol ρ	Deposit Functions (Selector(s))	Withdrawal Functions (Selector(s))	t_ρ	d_ρ	w_ρ	a_ρ	δ_ρ	Δ_ρ	$\lvert X_p \rvert$
Curve `0x8301AE4fc9c624d1D396cbDAa1ed877821D7C511`	`add_liquidity` (`0x0b4c7e4d`, `0xee22be23`)	`remove_liquidity` (`0x5b36389c`, `0x269b5581`) `remove_liquidity_one_coin` (`0xf1dc3cc9`, `0x8f15b6b5`)	1505	967	538	548160	4	1287105	806
Sturdy `0x9f72DC67ceC672bB99e3d02CbEA0a21536a2b657`	`deposit` (`0xe8eda9df`) `depositYield` (`0xd6996185`)	`withdraw` (`0x69328dec`) `withdrawFrom` (`0x12ade5ad`)	187	104	83	432472	18	1152668	93
Yearn V2 DAI `0xdA816459F1AB5631232FE5e97a05BBBb94970c95`	`deposit` (`0xd0e30db0`, `0xb6b55f25`)	`withdraw` (`0x3ccfd60b`, `0x2e1a7d4d` `0x00f714ce`, `0xe63697c8`)	322	10	312	503000	1081	1271917	226
Yearn V2 WETH `0xa258C4606Ca8206D8aA700cE2143D7db854D168c`			341	5	336	548424	26	1277893	266

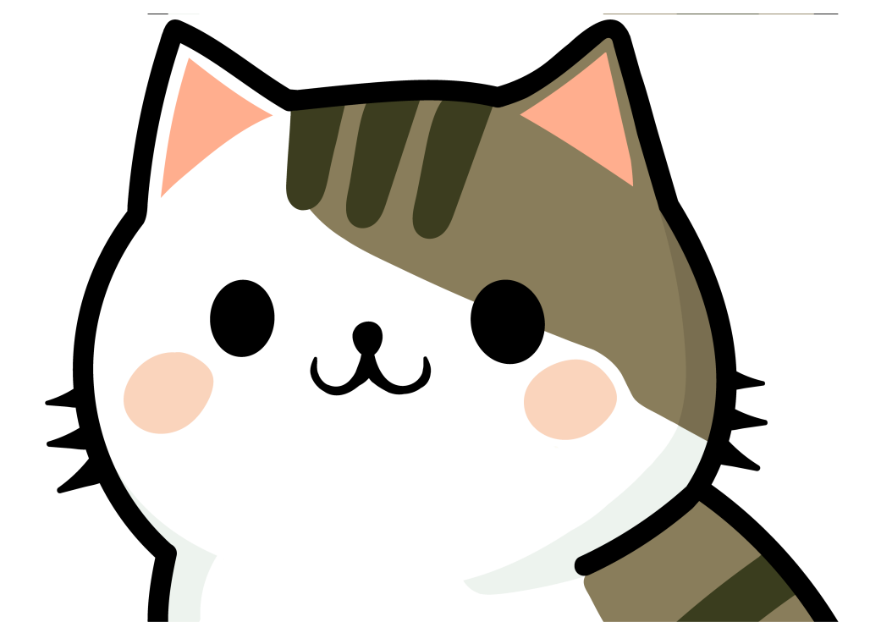
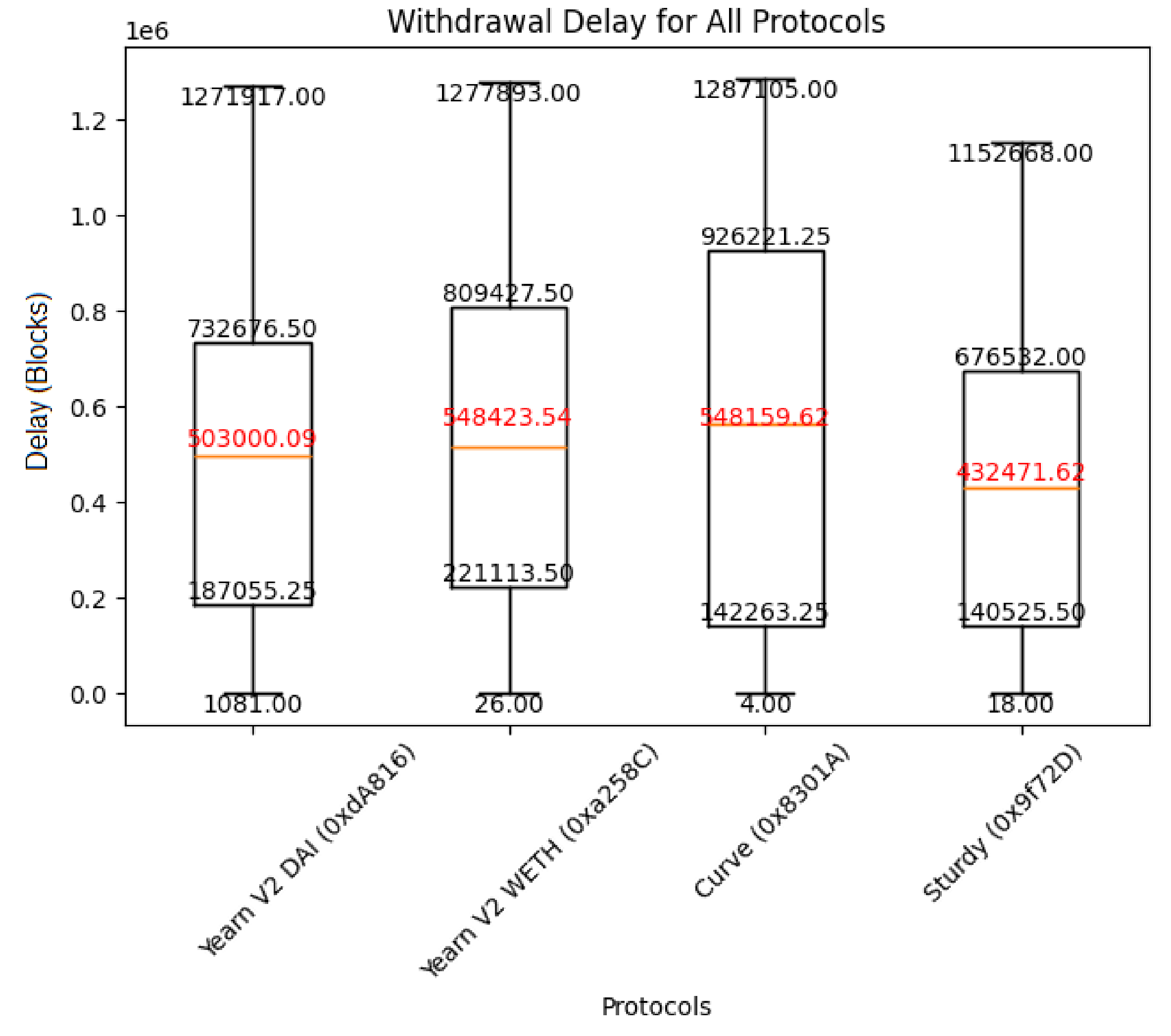
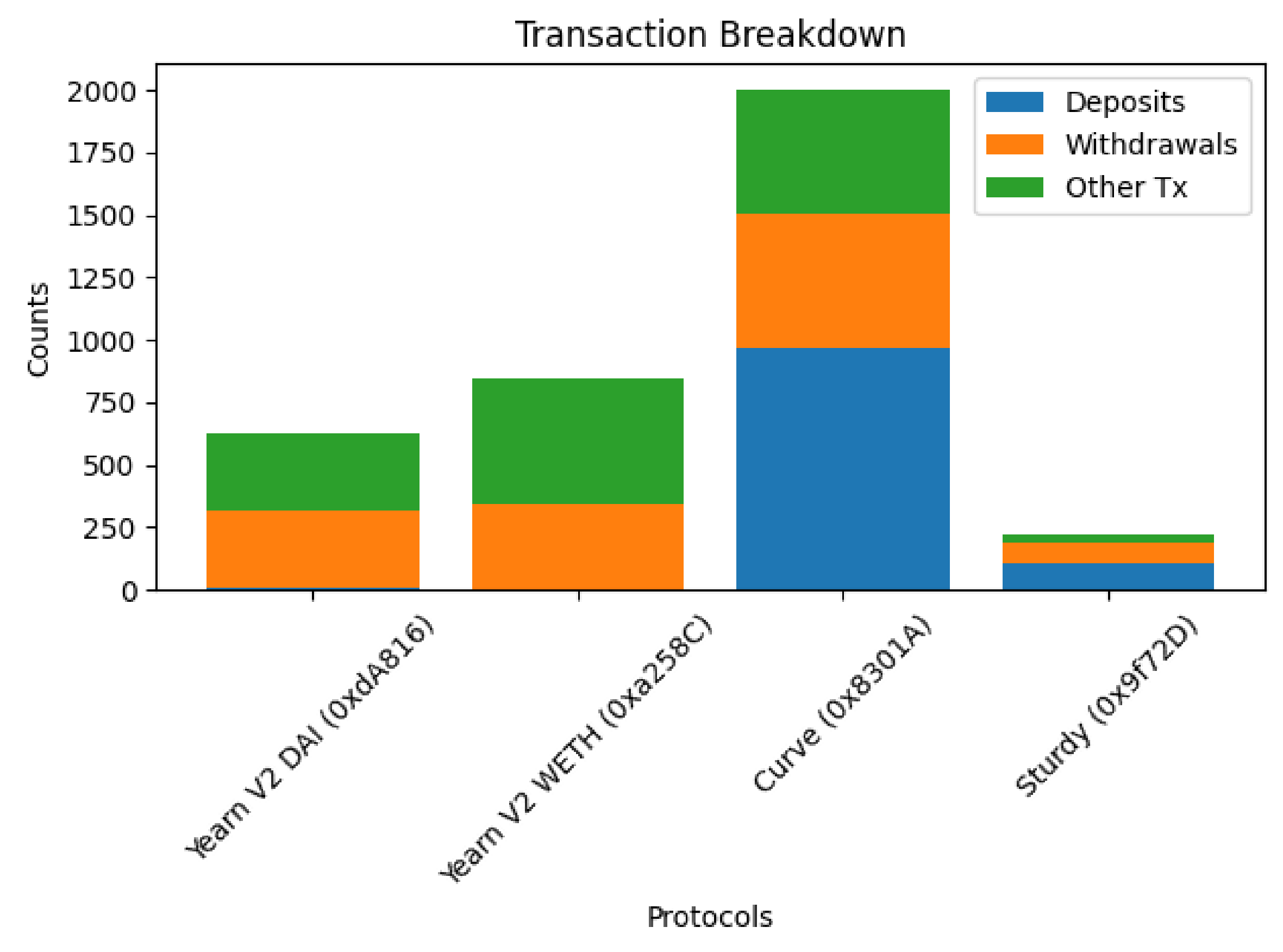


Table 1. Deposit delay statistics for the protocols studied in this work. For each protocol, we list the address of the Ethereum smart contract that has the corresponding deposit and withdrawal functions implemented. Each function is listed with its function selector (and may have multiple if the function name is overloaded).

Charts



(a) Box-whisker plot for withdrawal delay on the studied protocols.



(b) Bar chart for the number of transactions targeting the protocol smart contracts. Most have roughly the same number of withdrawals as deposits.

Figure 1. Visualization of findings.

Discussion (Continued)

Figure 1a shows the distribution of the delays in a box-whisker plot. The middle line in the box shows the median delay, while the middle text shows the mean delay. The upper and lower lines are the upper and lower extremes, respectively, and the box indicates the upper and lower quartile for delays. Most delays are around the median and mean; most users leave funds in for several days if not several weeks for these protocols.

Figure 1b visualizes the number of deposits, withdrawals, and other transactions. It is clear that both Yearn pools, and Curve, are more popular than Sturdy during the studied time period. Most protocols have roughly the same amount of deposits as withdrawals, though there are many other calls to Curve.

Threats to Validity

- **Incomplete data.** The work does not analyze the entire history of the blockchain, so some recorded gaps may be underestimated. Moreover, protocols may have funds deposited or withdrawn through intermediate smart contracts; these transactions are not included.
- **Imprecise deposit and withdrawal pairs.** The data was collected such that a user could deposit via any of the protocol's listed deposit functions and withdraw via any listed withdrawal function; this may not be accurate. There are also cases where a user withdraws someone else's deposit which are excluded.

Conclusion

We studied the delays between withdrawals and deposits for four Ethereum DeFi protocols. These were chosen because they suffered exploitation last year, and we studied the deposits and withdrawals of these protocols prior to their exploitation. Our data suggests that a mandatory delay between deposits and withdrawals would not have negatively affected users who interact directly with these protocols but such delays may have prevented exploitation of these protocols. Our work does not capture the full picture and future work is necessary to counter the threats to validity we outlined. Nonetheless, we stress that mandatory delays are a powerful tool in protecting protocol funds that developers should consider using.

References

[1] Valerian Callens, Zeeshan Meghji, and Jan Gorzny. Temporarily restricting solidity smart contract interactions. In *2024 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, pages 1–8, 2024. doi: 10.1109/DAPPS61106.2024.00008.

[2] Krzysztof Gogol, Christian Killer, Malte Schlosser, Thomas Bocek, Burkhard Stiller, and Claudio J. Tessone. SoK: Decentralized finance (DeFi) - fundamentals, taxonomy and risks. *CoRR*, abs/2404.11281, 2024. doi: 10.48550/ARXIV.2404.11281. URL `https://doi.org/10.48550/arXiv.2404.11281`.

[3] Stefan Kitzler, Friedhelm Victor, Pietro Saggese, and Bernhard Haslhofer. Disentangling decentralized finance (DeFi) compositions. *ACM Trans. Web*, 17(2), March 2023. ISSN 1559-1131. doi: 10.1145/3532857. URL `https://doi.org/10.1145/3532857`.

[4] Kaihua Qin, Liyi Zhou, Benjamin Livshits, and Arthur Gervais. Attacking the DeFi ecosystem with flash loans for fun and profit. In Nikita Borisov and Claudia Díaz, editors, *Financial Cryptography and Data Security - 25th International Conference, FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part I*, volume 12674 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2021. doi: 10.1007/978-3-662-64322-8_1. URL `https://doi.org/10.1007/978-3-662-64322-8_1`.

[5] Liyi Zhou, Xihan Xiong, Jens Ernstberger, Stefanos Chaliasos, Zhipeng Wang, Ye Wang, Kaihua Qin, Roger Wattenhofer, Dawn Song, and Arthur Gervais. SoK: Decentralized finance (DeFi) attacks. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, pages 2444–2461. IEEE, 2023. doi: 10.1109/SP46215.2023.10179435. URL `https://doi.org/10.1109/SP46215.2023.10179435`.