SoK: Decentralized Finance (DeFi)

Sam Werner
Imperial College London
London, United Kingdom

Daniel Perez
Imperial College London
London, United Kingdom

Lewis Gudgeon
Imperial College London
London, United Kingdom

Ariah Klages-Mundt
Cornell University
Ithaca, United States

Dominik Harz
Interlay
London, United Kingdom

William J. Knottenbelt
Imperial College London
London, United Kingdom

ABSTRACT

Decentralized Finance (DeFi), a blockchain powered peer-to-peer financial system, is mushrooming. Two years ago the total value locked in DeFi systems was approximately 700m USD, now, as of April 2022, it stands at around 150bn USD. The frenetic evolution of the ecosystem has created challenges in understanding the basic principles of these systems and their security risks. In this Systematization of Knowledge (SoK) we delineate the DeFi ecosystem along the following axes: its primitives, its operational protocol types and its security. We provide a distinction between technical security, which has a healthy literature, and economic security, which is largely unexplored, connecting the latter with new models and thereby synthesizing insights from computer science, economics and finance. Finally, we outline the open research challenges in the ecosystem across these security types.

KEYWORDS

Decentralized Finance, DeFi, Ethereum, Blockchain

ACM Reference Format:
Sam Werner, Daniel Perez, Lewis Gudgeon, Ariah Klages-Mundt, Dominik Harz, and William J. Knottenbelt. 2022. SoK: Decentralized Finance (DeFi). In *4th ACM Conference on Advances in Financial Technologies (AFT '22), September 19–21, 2022, Cambridge, MA, USA.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3558535.3559780

1 DEFI: FINANCE 2.0?

Consider two views on the promise of Decentralized Finance (DeFi). For the DeFi Optimist, DeFi amounts to a breakthrough technological advance, offering a new financial architecture that is non-custodial, permissionless, openly auditable, (pseudo)anonymous, and with potentially new capital efficiencies. According to this view, DeFi generalizes the promise at the heart of the original Bitcoin whitepaper [109], extending the innovation of non-custodial transactions to complex financial operations. In contrast, the DeFi Pessimist is concerned that, inter alia, the unregulated, hack-prone DeFi ecosystem serves to facilitate unfettered and novel forms of financial crime. The pseudo-anonymous nature of DeFi permits cryptocurrency attackers, scammers, and money launderers to move, clean, and earn interest on capital. A critical part of the debate between the DeFi Optimist and the DeFi Pessimist, but outside of the scope of this paper, is *moral* in nature. This SoK leaves this important facet aside, focusing instead on synthesizing and evaluating the technical innovations of DeFi, seeking to allow newcomers to the field to discover the essential features and problems of the DeFi terrain.

DeFi, in its ideal form, exhibits four properties. DeFi is:

(1) Non-custodial: participants have full control over their funds at any point in time
(2) Permissionless: anyone can interact with financial services without being censored or blocked by a third party
(3) Openly auditable: anyone can audit the state of the system, e.g., to verify that it is healthy
(4) Composable: its financial services can be arbitrarily *composed* such that new financial products and services can be created (similar to how one is able to conceive new Lego models based on a few basic building blocks)

DeFi has grown rapidly, going from around 700m USD in total value locked (TVL, or analogously "assets under management") at the start of 2020 to over 150bn USD as of April 2022. Ethereum alone accounts for 75bn USD in TVL, with the most capitalized use cases being collateralized lending, constituting c.54%, of the TVL, and decentralized exchange (DEXs), constituting c.31% of the TVL as of April 2022 [48]. In turn, this rise led to the 24 hour volume on a decentralized cryptoasset exchange [156], overtaking that of a major centralized cryptoasset exchange [34] for the first time [69].

Yet, as with any nascent technology, DeFi is not without its risks. The decentralized nature of DeFi necessarily allows any actor to write *unaudited* and even malicious smart contracts, where user funds can be lost through programming error or stolen. Moreover, the audit process itself is no guarantee of safety, with many *audited* protocols (e.g., [10, 75, 101, 138, 166]) suffering serious exploits.

This paper. For DeFi to fulfill the vision of the DeFi Optimist, it must first be secure. The central contribution of this SoK is to cleanly and exhaustively delineate the DeFi security challenge into technical security and economic security. The delineation centers on atomicity: whether the attack is near-instantaneous and can costlessly fail (and therefore risk-free), or has a non-instantaneous duration and where failure comes with a cost. This categorization has the benefit of cleanly mapping different types of models to each type of security and clarifying previously vague terms of

AFT '22, September 19–21, 2022, Cambridge, MA, USA

ACM ISBN 978-1-4503-9861-9/22/09…$15.00
https://doi.org/10.1145/3558535.3559780

"economic risk" that have been commonly misapplied to exploits that are better understood as technical in nature (e.g., DEX sandwich attacks). Prior to this paper, economic security risks were largely unexplored, in part because they require synthesizing insights from across computer science, economics, and finance.

This SoK is structured as follows. We outline DeFi primitives in Sec. 2 and systematize existing DeFi protocols by six types of operations in Sec. 3. We provide a definition of DeFi exploits in Section 4. We then define a novel functional categorization of technical and economic security risks in DeFi and classify different attacks for each security type in Sections 5 and 6, respectively. We then propose a set of six primary open research challenges for DeFi going forward that build on these security types in Sec. 7.

Related work. Several surveys and other SoKs exist on specific DeFi protocol types.[1] We direct the reader, when appropriate, to such SoKs for further material (e.g., [15, 41]). One well-structured survey on DeFi has been published after our pre-print [144], which categorizes DeFi protocols but focuses on high level risk and regulatory challenges. None of these works delineate the new types of security challenges across DeFi, which is the main focus of our SoK.

2 DEFI PRIMITIVES

DeFi protocols require an underlying distributed ledger such as a blockchain, a peer-to-peer distributed append-only record of transactions. We take the underlying distributed ledger layer solely as an input into DeFi and refer the reader to existing work (notably [14, 25, 71, 171]) for a fuller exposition of the blockchain layer itself. We assume that the ledger has the basic security properties of consistency, integrity and availability [175]. Without these security properties, DeFi protocols built on top of such a ledger would themselves become inherently insecure.

In this section, we draw attention to features of the underlying blockchain layer that are most pertinent to DeFi.

2.1 Smart Contracts and Transactions

Smart Contracts. The most important provision is that the underlying ledger offers the ability to use smart contracts, which are program objects that live on the blockchain. These are able to communicate with one-another, via message-calls, within the same execution context and support *atomicity*, i.e., a transaction either succeeds fully (state update) or fails entirely (state remains unaltered), such that no execution can result in an invalid state. In some cases, *bundles* of transactions, in which transactions are grouped together in a given sequence, can also be executed atomically (e.g., [61]).

Smart contracts rely on blockchains that are transaction-based state machines, whereby an agent can interact with smart contracts via transactions. Once a transaction is confirmed, the contract code is run by all nodes in the network and the state is updated. The underlying cost to state updates comes in the form of transaction fees charged to the sender. For instance, the Ethereum Virtual Machine (EVM) [165] on the Ethereum [27] blockchain is a stack machine which uses a specific set of instructions for task execution. The EVM maintains a fixed mapping of how much gas, an Ethereum-specific unit that denominates computational cost, is consumed per instruction. The total amount of gas consumed by a transaction is then paid for by the sender [121, 162].

In order for DeFi protocols to function on top of them, smart contracts must:

- be expressive enough to encode protocol rules
- allow conditional execution and bounded iteration
- be able to communicate with one-another, via message-calls, within the same execution context (typically a transaction)
- support atomicity, i.e., a transaction either succeeds fully (state update) or fails entirely (state remains unaltered), such that no execution can result in an invalid state

These properties provide *composability*, where smart contracts can be snapped together like Lego bricks ("Money Lego" [47]), with the possibility of building complex financial architectures. This is similar to as was envisaged in [82]. While promising, the side-effects of smart contracts interactions and the space of all possible interactions is vast. In a setting focused on financial applications, such complexity brings with it a great burden to understand the emergent security properties of composed smart contracts. We discuss this in more detail in Sections 5 and 6. One particularly common use of smart contracts is to implement tokens, on-chain assets.

Transaction Execution. When a blockchain network participant wishes to make a transaction, the details of the unconfirmed transaction are first broadcast to a network of peers, validated, and then stored in a waiting area (the *mempool* of a node). This mempool is then propagated among the network nodes. Participants of the underlying ledger responsible for ensuring consensus, *miners*, then choose which transactions to include in a given block, based in part on the transaction fee attached to each transaction. Transactions in a block are executed sequentially in the order in which the *miner* of the respective block included them. For a detailed treatment of how this process works, we refer the reader to [109, 110, 123, 165]. Miners have the ability to control the sequence in which transactions are executed. Hence, miners can order transactions in ways that will earn them revenues and even insert their own transactions to extract further revenues. Miners can even be bribed to undertake such transaction re-ordering [107, 164]. The value that miners can extract is known as *Miner Extractable Value* (MEV) [46]. We consider these issues in detail in Sec. 6.2.

2.2 Keepers

Protocols may rely on their on-chain state being continually updated for their security. In transaction-based systems, updating the on-chain state requires transactions that are triggered externally. Since smart contracts are not able to create transactions programmatically, protocols must rely on external entities to trigger state updates. These entities, *keepers*, are generally financially incentivized to trigger such state updates. For instance, if for whatever reason a protocol requires a user's collateral to be automatically liquidated under certain conditions, the protocol will incentivize keepers to initiate transactions to trigger such liquidation.

[1] Most have been pre-printed since our paper was originally released.

2.3 Oracles

An oracle is a mechanism for importing off-chain data into the blockchain virtual machine so that it is readable by smart contracts. This includes off-chain asset prices, such as ETH/USD, as well as off-chain information needed to verify outcomes of prediction markets. Oracles are relied upon by various DeFi protocols (e.g. [2, 97, 104, 125, 146]).

Oracle mechanisms differ by design and their risks, as discussed in [90, 100]. A centralized oracle requires trust in the data provider and bears the risk that the provider behaves dishonestly should the reward from supplying manipulated data be more profitable than from behaving honestly. Decentralized oracles offer an alternative. As the correctness of off-chain data is not verifiable on-chain, decentralized oracles tend to rely on incentives for accurate and honest reporting of off-chain data. However, they come with their own shortcomings. We provide a detailed overview of oracle manipulation risks and on the shortcomings of on and off-chain oracles in Sections 5.2 and 6.4.

2.4 Governance

Governance refers to the process through which a system is able to effect change to the parameters which establish the terms on which interactions between participants within the system take place [90]. Such changes can be performed either algorithmically or by agents. While there is existing work on governance in relation to blockchains more broadly (e.g. [16, 94, 130]), there is still a limited understanding of the properties of different mechanisms that can be used both for blockchains and DeFi.

Presently, a common design pattern for governance schemes is for a DeFi protocol to be instantiated with a benevolent dictator —sometimes distributing power over a small council or "multisig"— who has control over governance parameters, with a promise made by the protocol to eventually decentralize its governance process. Such decentralization of the governance process is most commonly pursued through the issuance of a governance token (e.g. [13, 36, 44, 105]), an ERC-20 token which entitles token holders to participate in protocol governance via voting on and possibly proposing protocol updates. This token represents ownership in a decentralized autonomous organization (DAO) that is tak
sed with stewardship of the protocol.

Protocol upgrades in the DAO setting come through proposals in the form of executable code, on which governance token holders vote. In order to propose protocol updates, the proposer has to hold or have been delegated a threshold number of governance tokens. For a protocol upgrade to be executed, a minimum number of votes is required, which is commonly called a "quorum" in this setting.

3 DEFI PROTOCOLS

We now present DeFi protocols categorized by the type of operation they provide (for an illustration see Fig. 1).

3.1 On-chain Asset Exchange

Decentralized exchanges (DEXs) [81, 99] are a class of DeFi protocol that facilitate the non-custodial exchange of digital assets, where all trades are settled on-chain and thus publicly verifiable. While DEXs initially only supported assets native to the chain on which they operate, wrapped tokens, such as wBTC [21] (wrapped Bitcoin), and novel cross chain solutions [52, 167, 171, 172] have enabled DEXs to overcome this limitation. Today, based on the mechanism for price discovery, DEXs come in different variants, such as *order book DEXs* (including individual [80, 161] and batch settlement [17, 68], see Appendix C for the latter) and *automated market makers* (AMMs) (e.g., [54, 106, 157]). Due to their widespread adoption and novelty in DeFi, we specifically focus on AMMs.

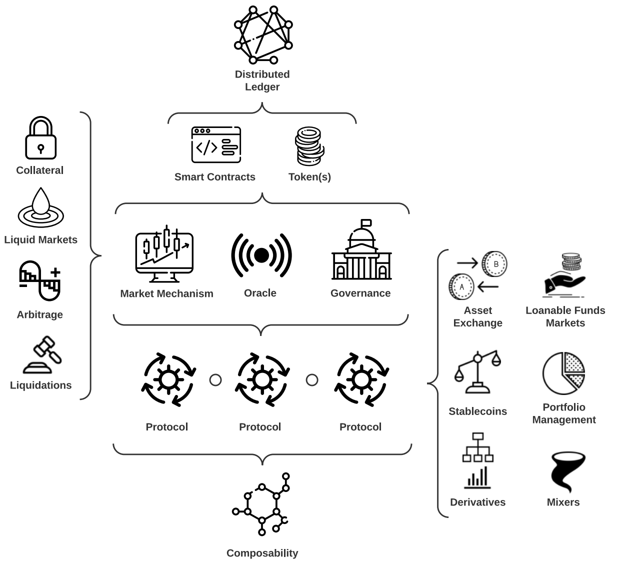

Figure 1: A conceptual overview of the different constructs within the DeFi ecosystem.

In traditional finance, market makers are liquidity providers that both quote a bid and ask price, selling from their own book, while making a profit from the bid-ask spread. Optimal market making strategies quickly become sophisticated optimization problems. In contrast, AMMs provide liquidity algorithmically through simple pricing rules with on-chain liquidity pools in place of order books and have been previously studied in algorithmic game theory, e.g., logarithmic market scoring rule (LMSR) [73] in prediction markets. While they have largely remained unimplemented in traditional finance, they have become popular in DeFi for a several reasons: (1) they allow easy provision of liquidity on minor assets, (2) they allow anyone to become a market maker, even if the market making returns are suboptimal, (3) AMM pools can be separately useful as automatically rebalancing portfolios, (4) maintaining an order book on-chain is inefficient.

In an AMM liquidity pool, reserves for two or more assets are locked into a smart contract, where for a given pool, each liquidity provider receives newly minted liquidity tokens to represent the share of liquidity they have provided. A trade is then performed by trading against a smart contract's liquidity reserve for an asset, whereby liquidity is added to the reserves of one token and withdrawn from the reserves of one or more other pool tokens. A trading fee is retained by a liquidity pool and paid out proportionally to the amount of liquidity provided by each liquidity token holder. Liquidity providers are required to give up their liquidity tokens in order to redeem their share of liquidity and accrued fees.

With an AMM, the price of an asset is deterministic and decided by a formula, not an order book, and thus depends on the relative sizes of the provided liquidity on each side of a currency pair. If the liquidity is thin, a single trade can cause a significant fluctuation in asset prices relative to the overall market, and arbitrageurs can profit by closing the spread. Arbitrage refers to the process of buying or selling the same asset in different markets to profit from differences in price. Parties who undertake this process are *arbitrageurs*, and play a critical role in DeFi protocols. Arbitrage is used to ensure that the price for an asset on an AMM is at parity with the price on the open market. Note that as the reserve ratios for a pool's assets change as liquidity is added and withdrawn, a liquidity provider may receive a different token ratio upon withdrawing his liquidity share compared to the ratio he initially deposited. For a more focused and formal analysis of AMM design and the underlying market making mechanism, we direct the reader to [3–6, 176].

3.2 Loanable Funds Markets for On-chain Assets

Lending and borrowing of on-chain assets is facilitated through *protocols for loanable funds* (PLFs) [72], which refer to DeFi lending protocols that establish distributed ledger-based markets for loanable funds of cryptoassets by pooling deposited funds in a smart contract. In the context of a PLF, a *market* refers to the total supplied and total borrowed amounts of a token, where the available deposits make up a market's liquidity. An agent may directly borrow against the smart contract reserves, assuming the market for the token is sufficiently liquid, where the cost of borrowing is given by the market's interest rate.

On PLFs, loans are generally of two forms: *over-collateralized loans* and *flash loans*. With an over-collateralized loan, a borrower is required to post collateral, i.e., provide something of value as security to cover the value of the debt, where the value of the collateral posted exceeds the value of the debt. In this way, collateralization simultaneously ensures that the lender (likely a smart contract) can recover their loaned value and provides the borrower with an incentive to repay the loan. In case the value of the locked collateral falls below some liquidation threshold, so-called liquidators, a type of keeper, are able to purchase the locked collateral at a discount and close the borrower's debt position [122].

An alternative to over-collateralized loans are flash loans. These are uncollateralized loans for the duration of a single transaction, requiring the borrower to repay the full borrowed amount plus interest by the end of the transaction. Flash loans leverage a blockchain's atomicity (i.e., the transaction fails if the loan is not repaid in the same transaction) and offer several use cases, such as decentralized exchange arbitrage and collateral swaps. However, they can also be used in attacks [127]. For a more detailed discussion and formal analysis of PLFs, we direct the reader to [15, 72].

3.3 Stablecoins

Non-custodial stablecoins are cryptoassets which aim to be price stable relative to a target currency, commonly the USD, and seek to achieve this via additional economic mechanisms. Note that custodial stablecoins, such as USDT [98] are not within the scope of DeFi, since these principally rely on a trusted third-party to operate, though they may be among the assets used in other DeFi protocols.

In the decentralized setting, the challenge for the protocol designer is to construct a stablecoin which achieves price stability in an economically secure and stable way and wherein all required parties can profitably continue to participate [90]. Price-stability is pursued via the use of on-chain collateral, providing a foundation of secured loans from which the stablecoin derives its economic value.

The core components of a non-custodial stablecoin are as follows [90].

- Collateral. This is the store of primary value for a stablecoin. Collateral can be exogenous (e.g., ETH in Maker [105]), where the collateral is primarily used externally to the stablecoin, endogenous (e.g., SNX in Synthetix [147]), where the collateral was created to be collateral or implicit (e.g., Nubits [95]), where the design lacks an explicit store of collateral.
- Agents. Agents form at least two roles in a non-custodial stablecoin: (1) risk absorption, for instance by providing collateral that is intended to absorb price risk, and (2) stablecoin users.
- Governance. A mechanism and set of parameters that governs the protocol as a whole (either performed by agents or algorithmically).
- Issuance. A mechanism to control the issuance of stablecoins against or using the collateral (either performed by agents or algorithmically).
- Oracles. A mechanism to import data external to the blockchain onto the blockchain, such as price-feeds.

See [90, 177] for a more complete discussion of stablecoin designs, models, and challenges.

3.4 Portfolio Management

For liquidity providers seeking to maximize their returns, liquidity allocation can be an onerous task given the complex and expansive space of yield-generating options. The management of on-chain assets can thus be automated through DeFi protocols which serve as decentralized investment funds, where tokens are deposited into a smart contract and an investment strategy that entails transacting with other DeFi protocols (e.g., PLFs) is encoded in the contract. Yield in DeFi is generated through interest (including accrued fees earned) and token rewards. For the latter, a protocol (e.g., PLF or AMM) distributes native tokens to its liquidity providers and/or users as rewards for the provision of deposits and/or protocol adoption. These protocol-native token rewards are similar to equity in the sense that they serve as a right to participate in the protocol's governance, as well as often represent a claim on protocol-generated earnings. The distribution model for token rewards in exchange for supplied liquidity may vary across protocols, yet is commonly proportional to how much liquidity an agent has supplied on a protocol. Therefore, smart contract-encoded investment strategies of on-chain assets are tailored around yield generating mechanisms of different protocols with the sole aim of yield aggregation and maximization. In practice, on-chain management of assets may range from automatic rebalancing of a token portfolio [59] to complex yield aggregating strategies [42].

3.5 Derivatives

Derivatives are financial contracts which derive their value from the performance of underlying assets. As of March 2022, the derivatives market represents about 62% of the entire cryptoassets trading market [43]. While about 99% of the derivative trading volume is achieved on centralized exchanges, a number of DeFi protocols have emerged which provide similar functionality [53, 113, 113, 163], with a particular focus on synthetic assets, futures, perpetual swaps and options. We lay out the adoption four different basic types of derivatives popular in the cryptoasset space[2]:

(1) **Synthetic assets.** In DeFi, synthetic assets typically replicate off-chain assets on-chain (e.g., the USD in protocols like Maker and Synthetix [147]). Though less used at present, another mechanism for constructing synthetic assets is to use AMMs that enact dynamic portfolio rebalancing strategies to replicate derivative payoffs. These bear a resemblance to synthetic portfolio insurance (see Ch. 13 in [78]) in traditional finance and have been explored more specifically using constant product market makers in [33, 57].

(2) **Futures.** Futures have seen little adoption in DeFi yet. Likely this is caused by the high volatility of the underlying cryptoassets making it hard to determine the risk taken by traders writing the futures.

(3) **Perpetual Swaps.** These are similar to futures, however, they have no set expiry date or settlement and were specifically created and popularized for cryptoasset markets [22]. Perpetuals allow traders to decide (typically on a daily basis, e.g., [53]) to keep the position by providing a funding transaction in case their position is underfunded. Due to the frequent price discovery, the price of perpetuals trades typically closer to the underlying in comparison to futures. Moreover, perpetuals are more capital efficient than trading the underlying itself since platforms require less than 100% collateral be posted by traders.

(4) **Options.** Currently, the DeFi market for options is very early with basic call and put options (e.g., [113, 163]). The cause for the limited adoption of options is three-fold. First, current option platforms are at least 100% collateralized. In comparison to their centralized counter-parts, this represents large capital inefficiency. Second, derivatives with set expiry dates like futures and options are hard to price on AMMs. Most AMM platforms (e.g., Uniswap [156]) do not account for a time dimension in the asset. This causes an issue specifically with option trading since the value of the option is subject to time decay. Possible remedies are more nuanced AMM designs like [111] that aim to incorporate such a time dimension. Also, complex value functions in the AMM like Balancer [13] allow replicating strategies that combine the underlying and a derivative into a single asset [57]. Third, options require a liquid market for efficient price discovery. Adoption will require solving the above problems to bootstrap the required liquidity that allows efficient pricing of those options.

3.6 Privacy-preserving Mixers

Mixers are methods to prevent the tracing of cryptocurrency transactions. These are important to preserve user privacy, as the transaction ledger is otherwise public information; however, this also means they could be used to obscure the source of illicit funds. Mixers work by developing a "shielded pool" of assets that are difficult to trace back before entering the pool. They typically take one of two forms: (1) mixing funds from a number of sources so that individual coins can't easily be traced back to address individually (also called a "coinjoin", e.g., [159]), or (2) directly shielding the contents of transactions using zero knowledge proofs of transaction validity (e.g., [153, 173]). Mixers serve as a DeFi-like application itself and additionally as a piece that could be included within other DeFi protocols.

4 WHAT IS A DEFI EXPLOIT?

We define what we consider an exploit to be in the DeFi context. To do this, we first need to set out a taxonomy of blockchain information. Let t denote the ordering of events. At any point t in this sequence of events, there are three categories of information relevant to a DeFi protocol:

- Off-chain ground truth, denoted $\mathcal{G}_t^{OFF}$. For example, true wind speed or the true equilibrium prices of assets traded in off-chain venues;
- On-chain ground truth, denoted $\mathcal{G}_t^{ON}$. For example, on-chain asset ownership;
- On-chain estimates of off-chain ground truth, denoted $\hat{\mathcal{G}}_t^{OFF}$. For example, oracle reported prices.

A smart contract only has access to $\mathcal{G}_t^{ON}$ and $\hat{\mathcal{G}}_t^{OFF}$. Now consider that a set of smart contracts (e.g., constituting a DeFi protocol), S, has a set of intended properties, P^S. Each intended property is a function of the information up until the current state in the sequence such that

$$P^S\left(\mathcal{G}_t^{ON}, \mathcal{G}_t^{OFF}\right) = \left\{P_0^S\left(\mathcal{G}_t^{ON}, \mathcal{G}_t^{OFF}\right), \dots, P_n^S\left(\mathcal{G}_t^{ON}, \mathcal{G}_t^{OFF}\right)\right\}.$$

For a particular property, it can either hold or not hold at any given state, as intended, such that the properties are Boolean. Some examples of these properties include the achievement of an invariant rule for an AMM or ensuring that debt positions remaining over-collateralized in accordance with a collateral factor in a Protocol for Loanable Funds (PLF).

For a smart contract to execute as intended, first, the smart-contract program must actually implement the intended logic (i.e., it must be free of implementation bugs). Second, since smart contracts often heavily depend on $\mathcal{G}_t^{OFF}$, on-chain estimates of off-chain information, $\hat{\mathcal{G}}_t^{OFF}$, must be sufficiently accurate to ensure

[2] For an introduction to derivatives, we refer the reader to [78]

that $P_i(\mathcal{G}_t^{ON}, \mathcal{G}_t^{OFF}) = P_i(\mathcal{G}_t^{ON}, \hat{\mathcal{G}}_t^{OFF})$. Such estimates can be manipulated: for example, smart contract states can experience 'flash' manipulation, where an artificial state is set up at the beginning of a transaction and unwound at the end of a transaction. Such manipulation can result in $\mathcal{G}_t^{OFF}$ and $\hat{\mathcal{G}}_t^{OFF}$ diverging.

The different categories of exploit now follow from this statement of what is required for a smart contract to reliably execute as intended. An exploit can arise from an attacker gaining, or seeking to gain from:

(1) Seizing upon a difference between the actual implementation of a smart contract and the intended implementation. See Section 5.1.
(2) Seizing upon an incidental difference between $\mathcal{G}_t^{OFF}$ and $\hat{\mathcal{G}}_t^{OFF}$. For example, in November 2020 an incidental deviation in the Dai price reported by Coinbase triggered liquidations on Compound[35].[3]
(3) Creating a deliberate difference between $\mathcal{G}_t^{OFF}$ and $\hat{\mathcal{G}}_t^{OFF}$. For examples, see 5.2 and 5.3.

In exploits of types (2) and (3), a protocol's operation is made to deviate from its intended properties, i.e., for some i, t,

$$P_i\left(\mathcal{G}_t^{ON}, \mathcal{G}_t^{OFF}\right) \neq P_i\left(\mathcal{G}_t^{ON}, \hat{\mathcal{G}}_t^{OFF}\right).$$

Note the importance of the notion of intention in this delineation of exploit types. For example, when prices on an AMM and prices off-chain differ, this does not amount to an exploit because an intended property of an AMM is that arbitrage will rebalance the AMM pool according to the AMM's pricing rule. However, when a protocol uses an AMM price as a price oracle, its operation is relying on information from $\hat{\mathcal{G}}_t^{OFF}$. When this information is significantly different from the true information in $\mathcal{G}_t^{OFF}$ (e.g., if an attacker manipulates the oracle price), this can cause the protocol to behave in unintended ways, which would amount to an exploit.

Exploits are usually worrisome when they are either profitable (i.e., an attacker can get more assets out of the exploit than they are spending to execute the exploit, usually from stealing assets) or when they cause high losses for protocols or users for low cost of execution (which itself is often profitable by short selling the token of a protocol to be exploited).

In the following two sections we delineate between two general types of exploits in a novel way that simultaneously distinguishes how the exploit is performed, the type of risks taken on by an attacker, and the types of tools and models required to analyze security in the two contexts. Note that some attacks on DeFi protocols may not be categorized as security exploits at all. For example, the collapse of the Terra stablecoin [24] (and related collapses like the Iron stablecoin before it [49]) may be better described as currency runs/attacks as opposed to security exploits considering that they result from breaching the economic limits of a mechanism without necessarily exploiting the formal properties of smart contracts.

5 TECHNICAL SECURITY

We define a DeFi security risk to be *technical* if an agent can atomically exploit a protocol. In a technical exploit, an attacker effectively finds a sequence of contract calls, whether in a single transaction or a bundle of transactions, that leads to a profit through a violation of a protocol's intended properties (as visualized in Fig. 2). Such exploits can be performed risk-free (and often in a sense 'instantaneously') because the outcomes for the attacker are binary: either the attack is successful and the attacker profits or the transaction reverts (effectively the attack doesn't happen) and the attacker only loses some gas fees.

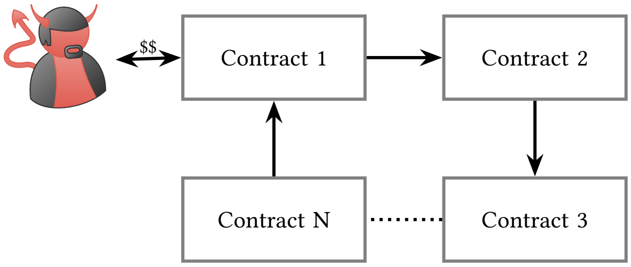

Figure 2: Diagram of a technical exploit.

In current blockchain implementations, this coincides with (1) manipulating an on-chain system within a single transaction, which is risk-free for anyone, and (2) manipulating transactions within the same block, which is risk-free for the miner generating that block or for an attacker who creates a bundle of transactions that are required to execute atomically in the order given. By exploiting technical structure, the underlying blockchain system allows no opportunity for markets or other agents to react in the course of such exploits (when such reaction is possible, we enter the domain of *economic* security problems in the next section). When there is competition to perform these exploits, they will factor into the game theory of blockchain mining (e.g., [20]) as part of MEV extraction (as discussed in [46]); however, attempting these exploits will be risk-free (minus potential gas fees) for any attacker. We identify three categories of attacks that fall within technical security risks of DeFi protocols: attacks exploiting smart contract vulnerabilities, attacks relying on the execution order of transactions in a block, as well as attacks which are executed within a single transaction. These security risks are often addressable with program analysis and formal models to specify protocols, although these problems can quickly become complex to formulate and computationally hard.

Technical Security

A DeFi protocol is technically secure if it is not possible for an attacker to atomically exploit the protocol at the expense of value held by the protocol or its users. Due to atomicity, these attacks can generate risk-free profit. A common property of technical exploits is that they occur within a single transaction or a bundle of transactions in a block.

Examples of past DeFi protocol exploits that fall into the presented attack categories of technical security are given in Table 1 in Appendix A.

[3] Where such incidental differences are not seized upon this might more properly be considered a system failure, but we group these together as both are relevant for analyzing protocol security

5.1 Smart Contract Vulnerabilities

Smart contract vulnerabilities have been extensively discussed in the literature [8, 120, 155] and we will therefore not give an extensive list of all the known vulnerabilities but rather focus on the one which have already been exploited in the DeFi context.

Reentrancy. A contract is potentially vulnerable to a reentrancy attack if it delegates control to an untrusted contract, by calling it with a large enough gas limit, while its state is partially modified [140]. A trivial example is a contract with a withdraw function that checks for the internal balance of a user, sends them money and updates the balance. If the receiver is a contract, it can then repeatedly re-enter the victim's contract to drain the funds. Although this attack is already very well-known, it has been successfully used several times against DeFi protocols[40, 51].

In practice, reentrancy vulnerabilities are generally simple to detect and fix by using static analysis tools [103, 155]. There are two main ways to prevent this vulnerability: (1) using a reentrancy guard that prevents any call to a given function until the end of its execution or (2) finalizing all the state updates before passing execution control to an untrusted contract.

Integer Manipulation. Almost every DeFi application manipulates monetary amounts in some way or another. This often involves not only adding and subtracting to balances but also converting into different units or to different currencies. We present the two most common types of integer manipulation issues.

The first issue, which has been extensively studied in the literature [86, 154], is integer over- and underflow. As the Ethereum Virtual Machine (EVM) [165] does not raise any exception in case of over- or underflow, this will often result in failed transactions and cause the smart contract to misbehave [120].

The second issue is unit error during integer manipulation. The main language used to develop DeFi applications on Ethereum at the time of writing is Solidity [62], which has a limited type system and no support for operator overloading. In addition, the EVM only supports a single type, 32 byte integers, and has no built-in support for fixed-point numbers. To work around this limitation, each protocol decides on an arbitrary power of 10 to use as its base unit, often 10^{18}, and all the computations are performed in terms of this unit. However, given the limitations of the type-system, most programs elect to use exclusively 32 byte integers. Arithmetic on two units accidentally on different scales would not be caught by the compiler.

Logical Bugs. There are a large number of exploits that are rooted in simple programming errors in the smart contracts. While logical bugs are by no means unique to smart contracts, but common to any type of software, the consequences for smart contracts, where immutability underpins the system, can be much more severe than for many other genres of software and result in unrecoverable financial losses.

A large share of the bugs found in Table 1 are also very simple mistakes that have been overlooked in both the development process and professional contract audits. We discuss in Section 7 potential mitigation techniques to these issues.

5.2 Single Transaction Attacks

We refer to attacks which can be successfully executed, independent of knowing about some other pending transaction, as single transaction attacks. This category of attack is leveraging transaction atomicity and composability of smart contracts.

Governance Attacks. Protocols that implement decentralized governance mechanisms tend to rely upon governance tokens, which empower token holders to propose and vote on protocol upgrades. The technical structure of these governance mechanisms can sometimes be exploited. For instance, many governance designs allow updates to be instantaneously proposed and approved and the protocol code upgraded within a single transaction. In this setting, an attacker may obtain an amount of governance tokens sufficient to propose and execute malicious contract code and steal a contract's funds, all while circumventing the usual governance process in which other participants can react [70]. The attacker may even obtain the share of governance tokens instantaneously within the same transaction (i.e., they may not be a long-term participant) through flash loans from PLFs and swaps from AMMs. In fact, large quantities of governance tokens can be obtained easily in these ways today, and such attacks have been executed in practice [102].

The direct problem can usually be sorted by applying a timelock to the governance process so that updates cannot be performed instantaneously and other participants have a chance to react. However, as we will see in the economic security section, this often does not solve the incentive issues completely, it just resolves the *technical* issue.

Single Transaction Sandwich Attacks. In a single transaction sandwich attack, an attacker manipulates an instantaneous AMM price in order to exploit a smart contract that uses that price. An attacker first creates an imbalance in an AMM, exploits composable contracts which rely on the manipulated price, and then reverses the imbalance to cancel out the cost of the first step. The whole sequence can be performed atomically in a single transaction risk-free. Creating an imbalance typically requires access to a large amount of capital. In a system with flash loans/minting, all agents effectively have such access, although we stress that these attacks are still possible for large capital holders regardless of whether flash loans/minting are widespread. In practice, this type of attack has occurred multiple times [118, 131]. One of the most prominent single transaction sandwich attacks in terms of seized funds was performed against the Harvest protocol [56]. The attacker took out a $50m USDT flash loan from Uniswap and used part of the funds to create an imbalance in the liquidity reserves of USDC and USDT on Curve [44] (an AMM) to increase the AMM's virtual price of USDT. As the price of USDT on Curve was used as an on-chain oracle by the Harvest protocol, the attacker was able to mint Harvest LP tokens (i.e., tokens a liquidity provider receives in exchange for depositing funds into a protocol) by depositing 60.6m USDT, before reversing the imbalance on Curve and withdrawing 61.1m USDT from Harvest. The attacker was able to withdraw more USDT than deposited, as at the time of the withdrawal, the USDT price given by Curve was less than the deposit price, and therefore one Harvest LP token was worth more USDT during withdrawal. The attacker repeated this attack 32 times, draining a total of $33.8m of the protocol's funds.

To protect against such manipulations, AMMs include a limit amount (or maximum slippage) that a trade can incur, though this only prevents manipulations above this amount.

5.3 Transaction Ordering Attacks

In traditional finance, the act of *front-running* refers to taking profitable actions based on non-public information on upcoming trades in a market. In the context of blockchain, front-running a transaction refers to submitting a transaction which is solely intended to be executed *before* some other pending transaction [55]. As transactions are executed sequentially according to how they have been ordered in a block, an agent may financially benefit from front-running one or more transactions, by having their transaction executed before a victim transaction. Similarly, an agent may pursue *back-running*, whereby a transaction is intended to be executed *after* some designated transaction. As the majority of Ethereum miners order transactions by their gas price [179], an agent can set a higher or lower gas price relative to some target transaction, in order to have his transaction executed before or after the target, respectively. In the case of multiple agents attempting to front-run the same transaction, front-running results in priority gas auctions [46], i.e. the competitive bidding of transaction fees to obtain execution priority.

We refer to attacks which involve front- and/or back-running within a single block, thereby undermining the technical security of DeFi protocols, as transaction ordering attacks. Note that an attacker does not need to be a miner in order to execute the following attacks but such attacks can be undertaken risk-free if the attacker is a miner.

Displacement Attacks. In a displacement attack, an attacker front-runs some target transaction, where the success of the attack does not depend on whether the target transaction is executed afterwards [55]. A simple example of such an attack would be an attacker front-running a transaction that registers a domain name [85]. A more severe risk comes in the form of *generalized* front-runners [139], which are bots that parse all unconfirmed transactions in the mempool, trying to identify, duplicate, modify and lastly front-run any transaction which would result in a financial profit to the front-runner. Examples of transactions vulnerable to generalized front-runners would be reporting a bug as part of a bug bounty scheme to claim a reward [26] and trying to 'rescue' funds from an exploitable smart contract [139, 143].

Multi-transaction Sandwich Attacks. In a "sandwich attack", an attacker alters the deterministic price on an AMM prior to and after some other target transaction has been executed in order to profit from temporary imbalances in the AMM's liquidity reserves. In simple cases (e.g., Uniswap), the instantaneous AMM price is simply a ratio of AMM reserves and imbalances can be created simply by changing this ratio (e.g., by providing single-sided liquidity or performing a large swap through the AMM). This is how these AMMs are designed to work: swaps create imbalances, which, if left unbalanced, incentivize arbitrageurs to perform the reverse actions to balance the AMM pool.

An attacker may target another user's transaction (e.g., to profit from triggering large slippage in another user's swap) by trying to place adjacent transactions that set up the imbalance right before the swap and close out the imbalance right after the swap [145, 179]. This can be achieved through front-running the user's swap transaction by setting a higher gas price on the transaction creating the imbalance. By setting a lower gas price on the transaction closing the imbalance, the attacker can back-run the user's transaction and complete the attack. Note that setting high and low transaction fees does not guarantee the attack to succeed, as ultimately it is up to a transaction's miner to determine the order of execution.

Protocol	Loss	Audit	Attack	Date	Ref.
bZx	0.35m	✓	TX sandwich	Feb-15-2020	[65]
bZx	0.63m	✓	TX sandwich	Feb-18-2020	[11]
Uniswap	0.30m	✓	Reentrancy	Apr-18-2020	[40]
dForce	25.00m	✗	Reentrancy	Apr-19-2020	[64]
Hegic	0.05m	✗	Logical bug	Apr-25-2020	[148]
Balancer	0.50m	✓	TX sandwich	Jun-28-2020	[1]
Opyn	0.37m	✓	Logical bug	Aug-04-2020	[114]
Yam	0.75m	✗	Logical bug	Aug-12-2020	[32]
bZx	8.10m	✓	Logical bug	Sep-14-2020	[10]
Eminence	15.00m	✗	TX sandwich	Sep-29-2020	[74]
MakerDAO	-	✓	Governance	Oct-26-2020	[102]
Harvest	33.80m	✓	TX sandwich	Oct-26-2020	[75]
Percent	0.97m	✓	Logical bug	Nov-04-2020	[119]
Cheese Bank	3.3m	✓	TX sandwich	Nov-06-2020	[126]
Akropolis	2.00m	✓	Reentrancy	Nov-12-2020	[166]
Value DeFi	7.00m	✗	TX sandwich	Nov-14-2020	[118]
Origin	7.00m	✓	Reentrancy	Nov-17-2020	[101]
88mph	0.01m	✓	Logical bug	Nov-17-2020	[116]
Pickle	19.70m	✗	Logical bug	Nov-21-2020	[149]
Compounder	10.80m	✓	Logical bug	Dec-02-2020	[63]
Warp Finance	7.80m	✓	TX sandwich	Dec-18-2020	[132]
Cover	9.40m	✓	Logical bug	Dec-28-2020	[138]
Yearn	11.00m	✗	TX sandwich	Feb-05-2021	[137]
Growth DeFi	1.30m	✓	Logical bug	Feb-09-2021	[133]
Meerkat	32.00m	✗	Logical bug	Mar-04-2021	[135]
Paid Network	27.00m	✗	Logical bug	Mar-05-2021	[136]
DODO	2.00m	✗	Logical bug	Mar-09-2021	[134]
Cream	130.00m	✓	TX sandwich	Oct-27-2021	[170]

Table 1: An overview of empirical technical security exploits in DeFi protocols for the period February 2020 to March 2021. The included exploits are explicitly limited to technical exploits and exclude any deliberate protocol scams that may have occurred. Note that the amount of funds seized per exploit is denominated in USD as of the time of the exploit and does not account for any losses that may have been recovered.

A variant of this attack [179] can be performed if instead of being a liquidity taker, the attacker is a liquidity provider for the respective AMM. The attacker can front-run a victim transaction that swaps token A for token B and remove liquidity, exposing the victim to higher slippage. Subsequently, the attacker can back-run the victim transaction, and resupply the previously withdrawn liquidity. In a third transaction that swaps B for A, the attacker obtains a profit in B. A formal analysis of sandwich attacks is given in [179].

6 ECONOMIC SECURITY

We define a DeFi security risk to be *economic* if an attacker can perform a strictly non-atomic exploit to realize a profit at the expense of value held by the protocol or its users. In an economic exploit, an attacker performs multiple actions at different places in the transaction sequence but doesn't control what happens between their actions in the sequence, which means that there is no guarantee that the final action is profitable (as visualized in Fig 3 and in comparison to the technical exploit in Fig 2). Economic security is effectively about an exploiting agent who tries to manipulate a market or incentive structure over some time period (even if short, it is not instantaneous). Compared to technical exploits, since economic exploits are non-atomic, they come with upfront tangible costs, a probability of attack failure and risk related to mis-estimating the market response. Thus they are not risk-free and commonly involve manipulations over many transactions or blocks.

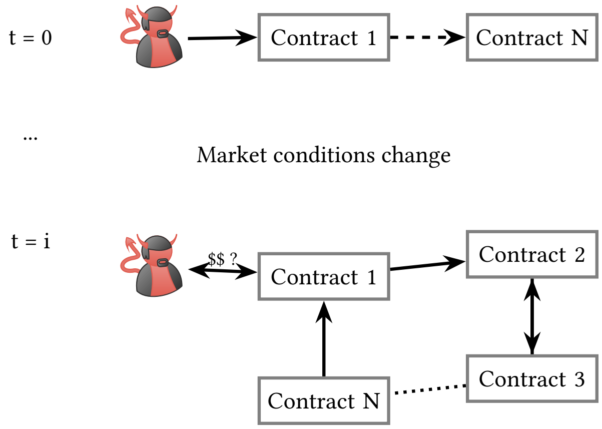

Figure 3: Diagram of an economic exploit.

In addition to comparing the structures in Figures 2 and 3, we provide a simple example to help illustrate the distinction between technical and economic security. Consider a protocol that uses an instantaneous AMM price as an oracle. An attacker can perform a (atomic) sandwich attack to steal assets, which amounts to a technical exploit. If instead the protocol used a time-weighted average AMM price as an oracle, then the attacker could manipulate this price over time (non-atomically) and may still be able to steal assets, which would amount to an economic exploit.

Economic risks are inherently a problem of economic design and cannot be solved by technical means alone. To illustrate, while these attacks could be performed atomically (and risk-free) in a very poorly constructed system that allowed it, they are not solved, for example, just by adding a time delay that ensures they are not executed in the same block. Even if all technical issues are sorted, we are often left with remaining economic problems about how markets or other incentive structures could be manipulated over time to exploit protocols. From a practical perspective, progress on these economic problems inherently requires economic models of these market equilibria and the design of better protocol incentive structures. These models differ considerably from traditional security models and are sparsely studied. As a result, defensive measures for economic security risks are also not as well established.

In this way also, technical security must be a first bar: if a protocol is not technically secure, then it will break in the presence of rational agents. Economic security only makes sense if technical security is achieved. For instance, if a protocol's funds can be exploited because it is not technically secure, then in an economic sense no rational agents should participate.

Economic Security

A DeFi protocol is economically secure if it is economically infeasible (e.g., unprofitable) for an attacker to perform exploits that are strictly non-atomic at the expense of value held by the protocol or its users. As economic exploits are non-atomic (or else they are better described as technical), they are not risk-free.

Economic Rationality. A central assumption in considering the class of economic security attacks is that of economic rationality. Following the standard game theoretic approach, we denote the strategy for player i as s_i. A strategy is a plan for what to do at each decision node (equivalently, information set) that the agent is aware they might reach. For example, a strategy would define what action an agent would take in the event that it finds itself in a protocol that becomes undercollateralized. A strategy $s_{1,i} \in \S_i$ for player i strictly dominates another strategy $s_{2,i} \in \S_i$ if regardless of the actions of other agents, strategy $s_{1,i}$ will always result in a higher payoff to the agent. Economic rationality is then defined as follows.

Economic Rationality

An agent is rational iff they will never play a strictly dominated strategy (i.e., they are profit optimizing).

Moreover, common knowledge of rationality means that all agents know no agent will play a strictly dominated strategy.

While most economic security analysis ought to consider attackers who have profit-maximizing objectives, it can also be important to consider attackers with other objectives. For instance, an attacker who wishes to shut down the system may decide to attack as long as the cost is of a moderate level. In this sense, the economic security depends on system interruptions being too costly to effect.

Incentive Compatibility. *Incentive compatibility* is originally a concept from game theory (e.g., [141], but as a concept has seen some adaption in the context of cryptoeconomics and in particular DeFi.

In the cryptoeconomic setting, a mechanism is defined as incentive compatible if agents are incentivized to execute the mechanism *as intended* (see e.g. [142]).

Cryptoeconomic Incentive Compatibility

A mechanism (or protocol) is incentive compatible iff agents are incentivized to execute the game as intended by the protocol designer.

A central question in the context of incentive compatibility, considered in [90], is the sustainability of the mechanism implemented by a system (i.e., will the incentives arising from the system allow the system to be economically secure and stable long-term). In [90], for stablecoins, this is separated into a question of incentive security, which is included in our concept of economic security, and a question of economic stability, which is a further question of whether an economically secure system actually plays out to the desired equilibrium envisioned by the designers.

We primarily focus on the direct security questions in this paper; however, similar questions to economic stability apply to protocols other than stablecoins as well. For instance, when designing synthetic derivatives built using dynamic portfolios (and implemented as AMM pools), a lingering question is how well these designs can replicate the derivative payoffs under extreme conditions. As a comparison, synthetic portfolio insurance in traditional markets can break down when markets move too fast for the strategy to rebalance (See Ch. 13 in [78]). AMM pools aim to rebalance over much shorter timescales, and so may have an advantage here, but are also suboptimal in other areas of rebalancing.

6.1 Overcollateralization as Security

Collateralization is one of the primary devices to ensure economic security in a protocol. In general, collateral serves as a potential repercussion against misbehaving agents [76] and allows creating protocols such as stablecoins, loanable funds, or decentralized cross-chain protocols. As asset prices evolve over time, these systems generally allow automated deleveraging: if an agent's level of collateralization (value of collateral / value of borrowing) falls below a protocol-defined threshold, an arbitrager in the system can reduce the agent's borrowing exposure in return for a portion of their collateral at a discounted valuation. This aims to keep the system fully collateralized.

Overcollateralization is not without risks, however. For instance, as explored in [70, 87], times of financial crisis (wherein there are persistent negative shocks to collateral asset prices) can result in thin, illiquid markets, in which loans may become undercollateralized despite an automated deleveraging process. In such settings, it can become unprofitable for liquidators, a type of keeper, to initiate liquidations. Should this occur, rational agents will leave their debt unpaid as that results in a greater payoff.

Another form of deleveraging risk arises when the borrowed asset has endogenous price effects, for instance when its price is affected by other agents' decisions in the system or when it is manipulable. This is the case in non-custodial stablecoins like Dai that are based on leverage markets (Dai is created by "borrowing" it against collateral and similarly must be returned to later release the collateral). As explored in [91, 92], such stablecoins can have deleveraging feedback effects that lead to volatility in the stablecoin itself. In regions of instability, the stablecoin will tend to become illiquid and appreciate in price (more so as they need to be purchased for liquidations), which can force speculative agents who have leveraged their positions to pay premium prices to deleverage. This causes their collateral to drawdown faster than may be expected, which makes the system in total less healthy and may lead to shortfalls in collateralization. This was later directly observed in Dai on "Black Thursday" [66]. As further discussed in [92], such a stablecoin requires uncorrelated collateral assets to be fully stabilized from such deleveraging effects as stable regions are related to submartingales (i.e., agents expect collateral asset prices to appreciate). However, current uncorrelated assets are primarily centralized/custodial, which poses a challenge for non-custodial designs.

6.2 Threats from Miner Extractable Value

An assumption by many blockchain protocols is that the block reward is sufficient to incentivize "correct" miner behavior. However, there are consensus layer risks should the MEV exceed the block reward. The simplest example of MEV is double spending of coins, which is commonly considered in base layer incentives. DeFi applications give rise to many new sources of MEV. For instance, (1) DEXs present atomic arbitrage opportunities between different trading pairs, as explored in [46], and (2) stablecoins built on leverage markets (like Dai) present arbitrage opportunities in liquidating leveraged positions, as explored in [91]. Similarly, other protocols, like PLFs, that utilize liquidation mechanisms also create MEV opportunities. Further, MEV can arise when miners are incentivized to re-order or exclude transactions based on cross-chain payments happening on other chains [83].

These are not exhaustive; there are additionally many other ways in which miners could manipulate DeFi protocols to extract value. It's worth noting that these are not just hypothetical concerns, they have actually been observed–e.g., [12, 23]–and shown to be sufficiently profitable, e.g., [178].

The practicality of MEV threats have been highlighted in [46], where the prevalent dangers of *undercutting* and *time-bandit* attacks are presented. In an undercutting attack [29], an adversarial miner would fork off a block with high MEV, while holding back some of the extractable value in order to incentivize other miners to direct their computational efforts towards the adversary's chain. In a time-bandit attack [46], an attacker forks from some previous block and sources *expected* MEV to increase his computational power and pursue a 51% attack until the expected MEV is realized. Hence, time-bandit attacks are a consensus layer risk and can be a direct consequence of historic on-chain actions which could profit a miner at some later point. A further threat is that miners could collude to set up more MEV opportunities over time, for instance by censoring transactions to top up collateral in crises and thus creating more liquidation events, as discussed in [91]. This is very similar to events on Black Thursday, in which mempool manipulations contributed to inefficient liquidation auctions in Maker [23].

6.3 Governance Risks and Governance Extractable Value

Protocol governance often introduces means to update system parameters and even redefine entire contracts. In many cases, this may be a necessary component for the system to evolve over time. However, governance can also introduce manipulation vectors that affect security. Governance of a DeFi protocol is typically tied to holders of governance tokens, which can often be thought of as shares in the protocol. In systems where there is large flexibility for governance to change the system, an important question is where governance token value comes from. A typical aim is for the protocol to incentivize good stewardship from its governance token holders by compensating governance with cashflows from the system. In this case, governance token value is derived from future discounted cashflows. Another possibility is that governance is directly aligned with underlying users–e.g., because they are the same.

However, if these incentives are not of sufficient size, then it may be more profitable for governance token holders to extract value in less desirable ways, which we term *governance extractable value* (GEV).[4] An example of GEV is for governors to effect changes to the protocol in ways that provide outside benefits to themselves but may be harmful to the overall system health. For instance, the Cream protocol governance added high levels of very risky collateral assets that they had an outside stake in, arguably to their benefit but against the interests of the protocol [129].

GEV also includes explicit governance attacks. An instance of an explicit attack was the governance takeover of the Build Finance DAO, where a malicious actor passed a proposal to take control of the Build token contract and was thereby not only able to drain various AMM pools by minting and swapping Build tokens, but to ultimately remove the DAO from any form of control over the core protocol [60]. A hypothetical GEV attack to indirectly extract collateral value is described in [89]. In cases like these, governance may not be incentive compatible. And if the value of governance tokens from incentive compatible sources crashes, the region of incentive compatibility also shrinks, and it may become profitable for a new coalition of governors to form to attack the protocol. This is increasingly problematic given the ease and low cost with which governance tokens may be obtained via flash loans and PLFs. Other complications arise in the need to protect minority rights within the protocol–e.g., building in limitations so that a majority of governors cannot unilaterally change the game to, for instance, steal all value of the other minority or users. See [96] for further discussion of GEV.

There is limited literature in modeling GEV incentives in the DeFi setting (as opposed to modeling governance in the underlying blockchain itself). The capital structure-like models developed in [79, 90] can be applied more generally to DeFi protocols to model governance security and incentive compatibility around these issues. As can be understood in those models, these issues essentially arise because there may not be outside recourse (e.g., legal) in the pseudo-anonymous setting to disincentivize attacks and manipulations compared to the (idealized) traditional finance setup. Further, [90] conjectures that in the case of a fully decentralized stablecoin with multiple classes of interested parties and with a high degree of flexibility for governance design, there exists no long-term incentive compatible equilibrium. Intuitively, there are resulting costs of anarchy in such systems, which can be too much to bear. In such a case, rational agents would choose not to participate. However, they also conjecture that other DeFi systems, such as DEXs, may have wider incentive compatibility in similar situations due to the different structure of such systems. These models have inspired new mechanisms such as Optimistic Approval [96], which provides an optional veto over governance updates to protocol users, as a new defensive measure to to reduce costs of anarchy and GEV.

6.4 Market and Oracle Manipulation

As the suppliers of off-chain information, oracles pose a fundamental component of DeFi protocols, particularly for sourcing price feeds [84]. However, it is important to distinguish between (1) a price that is manipulated yet correctly supplied by an oracle and (2) an oracle itself being manipulated. While we present each form of manipulation, note that the latter can be essentially modeled as a separate governance-type risk as discussed in [90].

Market Manipulation. We wish to quantify economic risks stemming from price manipulations in underlying markets while assuming the oracle follows a best practice implementation and is non-malicious. An adversary may manipulate the market price (on-chain or off-chain) of an asset over a certain time period if a profit can be realized as a consequence of the price manipulation–e.g., by taking positions in a DeFi protocol that uses that market price as an oracle. As discussed in the Section 5, instantaneous AMM prices are easily manipulable with near zero cost and, as a result, should not be used as price oracles. Market manipulation problems persist even when we assume the oracle is not an instantaneous AMM price. In this case, there is a cost to market manipulation related to maintaining a market imbalance over time, whether in an AMM (e.g., to manipulate a time-weighted average price) or through filling unfilled orders in an order book. Depending on whether the market for an asset is thick or thin, the cost for an attacker to significantly change the asset's price will be higher or lower, respectively. An instance of this form of market manipulation was seen with Inverse Finance, where a malicious actor first manipulated the used Sushiswap price oracle to quote a higher price for the Inverse token, only to exploit the protocol a block later by borrowing various assets against the Inverse token using the manipulated price before MEV bots arbitraged the manipulated price back [19]. A further example of such an attack would be to trigger liquidations by manipulating an asset's price, as discussed in the context of stablecoins in [91]. An attacker could profit either by purchasing liquidated collateral at a discount or shorting the collateral asset by speculating on a liquidation spiral. Such attacks are similar to short-squeezes in traditional markets. However, unlike with single transaction sandwich attacks, the aforementioned attack is not risk-free and could bring substantial losses to the attacker should it fail. In particular, markets and agents may react to such attacks in unpredictable ways.

To illustrate the potential of such attacks, the stablecoin DAI, which historically has thin liquidity, traded at a temporary price of $1.30 over a course of about 20 minutes on Coinbase Pro, a major centralized cryptoasset exchange, before returning to its intended $1 peg [88]. As a result, the Compound Open Price Feed [37], a cryptoasset price oracle which is in part based on prices signed by Coinbase, reported a DAI price of $1.23 to Compound for a short period of time. This incident triggered (arguably wrongful) liquidations on collateral worth approximately $89m, costing the liquidated Compound borrowers 23% (from the imbalanced DAI price) plus an additional 5% (the Compound liquidation incentive, i.e., the discount at which collateral is sold at during a liquidation) on their liquidated assets.

[4] GEV may be interpreted as a generalization of MEV with miners being a specific type of governor tasked with ordering transactions on the base layer.

Oracle Manipulation. Centralized oracles serve as a single point of failure and despite trusted execution environments [174] they remain vulnerable to the provider behaving maliciously if incentives are sufficient for manipulating the source of a data feed. Decentralized price oracles may use on-chain data, most notably on DEXs (specifically AMMs) for crypto-to-crypto price data. However, as outlined in Section 5.2, prices may be manipulable through intentionally created imbalances and thinly traded markets. Furthermore, on-chain DEX oracles inherently can not price off-chain assets and fiat currencies.

As discussed in [90], decentralized oracle solutions for off-chain data exist. However, they are yet imperfect solutions, tending to rely on Schelling point games, in which agents vote on the correct price values and are incentivized against having their stake slashed if their vote deviates from the consensus. Tying incentives to consensus, when the correctness of the consensus decision is not objectively verifiable (as in this case), paves a vector for game theoretic attacks, like in Keynesian beauty contests.

7 OPEN RESEARCH CHALLENGES

There are many open research challenges in DeFi stemming from the technical and economic security issues presented in Sections 5 and 6.

7.1 Composability Risks

Cryptoassets can be easily and repeatedly tokenized and interchanged between DeFi protocols in a manner akin to rehypothecation. This offers the potential to construct complex, inter-connected financial systems, yet bears the danger of exposing agents to composability risks, which are as of yet mostly unquantified. An example of composability risk is the use of flash loans for manipulating instantaneous AMMs and financially exploiting protocols that use those AMMs as price feeds. This has repeatedly been exploited in past attacks (e.g. [75, 126, 150]). However, the breadth of composability risks spans far beyond the negative externalities stemming from instantaneous AMM manipulations. For instance, there remain open questions about the consequences of the following types of exploitations on connecting systems: the accumulation of governance tokens to execute malicious protocol updates, the failure of non-custodial stablecoin incentives to ensure price stability, and failure of PLF systems to remain solvent. Note, however, that this list is far from exhaustive. These become increasingly important issues as more complex token wrapping structures [158] stimulate higher degrees of protocol interconnectedness. For example, the use of PLF deposit tokens (as opposed to the tokens in their original forms) within AMM pools and strategies to earn yield on underlying assets through leverage by borrowing non-custodial stablecoins and depositing into PLFs or AMMs.

Recent works [72, 108] begin to explore protocol interdependence, while [152] propose a process-algebraic technique that allows for property verification by modeling DeFi protocols in a compositional manner. Nonetheless, a critical gap in DeFi research toward taxonimizing and formalizing models to quantify composability risks remains. This problem is elevated as a holistic view on the integrated protocols is necessary: failures might arise from both technical and economic risks. Ensuring safety of protocol composition will be close to impossible for any protocol designer and forms a major challenge for DeFi going forward.

7.2 Governance

We identify important research directions in governance and GEV. A general direction is modeling incentive compatibility of governance in various systems with GEV. For instance, setting up models, finding equilibria, and understanding how other agents in the system respond. The models in [90] get this started in the context of stablecoins and additionally discuss how to extend to other DeFi protocols. There is moreover a range of discussions around simulating and formalizing governance incentives through tools like cadCAD [112].

There remain unanswered questions with regards to the general design of governance incentives. For instance, how to structure governance incentives to reward good stewardship: e.g., intrinsic vs. monetary reward, reward per vote vs. reward per token holder, and measures of good stewardship. Furthermore, there lies potential in formally evaluating protection of minority agents in systems with flexible governance and large GEV.

For systems utilizing governance tokens, we identify research gaps rooted in security risks of the ability to borrow governance tokens via flash loans and PLFs. Specifically, we identify opportunities to formally explore how (1) technical security can be compromised and (2) from an economic security point of view, incentive compatibility is further complicated by the borrowing of governance tokens.

7.3 Oracles

We highlight a few open challenges about oracle design and security. Note that, in many cases, the oracle problem can also be directly related to the governance problem, as governors are often tasked with choosing the oracles that are used.

A more general open challenge lies in how to structure oracle incentives to maintain incentive compatibility to report correct prices. This is similar to governance design in some ways and needs to take into account the possible game theoretic manipulations that could be profitable.

We identify a further research opportunity in designing and evaluating the security of various oracle strengthening methods. While there exist several works examining oracle designs on both a general and empirical basis–e.g. [84, 100]– a formal security analysis of, e.g., medianizers, reputation systems, and grounding reported prices based on on-chain verifiable metrics is yet to be done.

7.4 Miner Extractable Value

While research on MEV and the extraction of it is being put forward [46], methodologies to quantify negative externalities of MEV–e.g., from wasted gas per block, upward gas price pressure–and the full extent of MEV opportunities remain scarce. For the latter, we conjecture that the miner's problem to optimize the MEV they extract in a block is NP-hard and additionally hard to approximate. To support this, it is quite easy to reduce a simplified version of the problem, in which the MEV of each transaction is fixed, to the knapsack problem. Note that while the knapsack problem is NP-hard, it is easy to approximate. In fact, we expect a more realistic version of the miner's problem to be harder than knapsack because the transaction ordering the miner chooses also changes the MEV of the transactions (i.e., swapping two elements might change their weight in knapsack).

With regards to quantifying extractable value, [9] introduce Clockwork Finance Framework, a novel framework that applies formal verification to reason about DeFi security properties with respect to the profit that is extractable by an actor in a given system. It should be noted that the definition of economic security as defined in [9] would be closer to what we refer to as technical security or atomic MEV and that modeling non-atomic MEV requires economic models of the underlying markets, as discussed in Section 6.

There are interesting questions regarding how the emergence of MEV opportunities endogenously affects agents' behavior within DeFi protocols. Models for this are started in the context of stablecoins in [90].

A further open challenge remains with respect to designing protective mechanisms against (1) consensus layer instability risks that are induced by high MEV incentives and (2) time bandit attacks that seek to rewrite the recent transaction history–for example, which could aim to trigger and profit from increased protocol liquidations. On this point, [91] suggests that oracle price validity could be tied to recent block hashes to prevent such reorderings from extracting the protocol value, though potentially with costs to the economic security of the protocol in other ways.

7.5 Program Analysis

There exists a large amount of work [77], both in academia [103, 124, 155] and industry [39, 58], to analyze smart contract bugs and vulnerabilities. While smart contracts analysis tools keep improving, the number and scale of smart contract exploits are showing no sign of decrease and are, on the contrary, becoming more frequent. Although program analysis tools are no silver bullet and cannot prevent all exploits, Table 1 and the discussed exploits in Section 5 and in Appendix B hint that there are some recurring patterns that could be automatically detected and prevented.

Current program analysis tools can mainly be divided into two categories: (1) fully automatic tools checking for program invariants and (2) semi-automated verification tools checking for user-defined properties [7, 31, 124]. While the latter allows to verify business logic in ways that are not fully automatable, they are typically non-trivial to setup and require knowledge of software verification, which limits their use to projects with enough resources. On the other hand, fully automatic tools, which can be very easily setup and ran, usually focus on checking properties of a single contract in isolation [38, 58, 154, 155], such as unchecked exceptions or integer overflows. However, they have not evolved yet to embrace the composable nature of smart contracts, which makes it impossible for such tools to reason about scenarios where the issue happens due to a change in something external to the smart contracts, such as a sudden change in a price returned by an oracle. Further, most tools reason very little about *semantic* properties of the smart contracts, such as how a particular execution path can influence ERC-20 token balances. We believe that improvements in these areas will allow auditors and developers to analyze and deploy their contracts with more confidence, reducing the number of technical security exploits.

7.6 Anonymity and Privacy

The anonymity and privacy of DeFi protocols is at present a significantly understudied area. There is a tension between user's privacy being valuable in itself, while at the same time helping malicious users to escape the consequences of their actions. At present, a large proportion of DeFi transactions occurs in protocols built on Ethereum, wherein agents at best have pseudoanonymity. This means that if an agent's real-world identity can be linked to an on-chain address, all the actions undertaken by the agent through that address are observable. While recent advances in zero-knowledge proofs [115, 160] and multi-party computations [18, 128] hold many promises, these technologies are yet to gain traction in the context of DeFi. One of the main friction points is the large computational cost of these technologies, which make them very expensive to use and deploy in the context of DeFi. A decrease of computational cost of the underlying blockchain will be key to how widely privacy-preserving technologies can be deployed by DeFi protocols.

8 CONCLUSION

In this SoK we have considered DeFi from two points of view, the DeFi Optimist and the DeFi Pessimist, and examined the workings of DeFi systematically. First, we laid out the primitives for DeFi before categorizing DeFi protocols by the type of operation they provide. After distinguishing between different types of information relevant to a DeFi protocol, we provided a working definition of an exploit. We established economic security on the same level as technical security and created a novel functional categorization of the different types of risk therein. Further, we provided clear definitions of these risks as well as insights into the types of models that are needed to understand and defend against these risks. Finally, we drew the attention to open research challenges that require a holistic understanding of both the technical and economic risks.

While DeFi may have the potential to creating a permissionless and non-custodial financial system, the opinion put forward by the DeFi optimist, the open technical and economic security challenges remain strong. The DeFi pessimist is, at least for now, on firm ground: solving these challenges in a robust and scalable way is a central challenge for researchers and DeFi practitioners. In the end, however, it is the blend between promise and challenge — the tension between the views of the DeFi optimist and the DeFi pessimist — that makes DeFi a worthwhile and exciting area for research.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their feedback and suggestions. This project received partial funding from EPSRC Standard Research Studentship (DTP) (EP/R513052/1), the Ethereum Foundation, the Brevan Howard Centre for Financial Analysis, Smart Contract Research Forum, and a Bloomberg Fellowship.

REFERENCES

[1] 1inch: Balancer pool with sta deflationary token incident (2020), https://1inch-

exchange.medium.com/balancer-hack-2020-a8f7131c980e

[2] AAVE: Aave: Protocol whitepaper v1.0 (2020), https://github.com/aave/aave-protocol/blob/master/docs/Aave_Protocol_Whitepaper_v1_0.pdf, accessed: 13-08-2020

[3] Angeris, G., Chitra, T.: Improved price oracles: Constant function market makers. Proceedings of the 2nd ACM Conference on Advances in Financial Technologies (2020)

[4] Angeris, G., Evans, A., Chitra, T.: When does the tail wag the dog? Curvature and market making. arXiv preprint arXiv:2012.08040 (2020)

[5] Angeris, G., Evans, A., Chitra, T.: Replicating market makers. arXiv preprint arXiv:2103.14769 (2021)

[6] Angeris, G., Kao, H.T., Chiang, R., Noyes, C., Chitra, T.: An analysis of uniswap markets. Cryptoeconomic Systems Journal (2019)

[7] Annenkov, D., Spitters, B.: Towards a smart contract verification framework in coq. arXiv preprint arXiv:1907.10674 (2019)

[8] Atzei, N., Bartoletti, M., Cimoli, T.: A survey of attacks on ethereum smart contracts (sok). In: International conference on principles of security and trust. pp. 164–186. Springer (2017)

[9] Babel, K., Daian, P., Kelkar, M., Juels, A.: Clockwork finance: Automated analysis of economic security in smart contracts. arXiv preprint arXiv:2109.04347 (2021)

[10] Baker, P.: Defi lender bzx loses $8m in third attack this year. CoinDesk (2020), https://www.coindesk.com/defi-lender-bzx-third-attack

[11] Baker, P.: Defi project bzx exploited for second time in a week, loses $630k in ether. CoinDesk (2020), https://www.coindesk.com/defi-project-bzx-exploited-for-second-time-in-a-week-loses-630k-in-ether

[12] Baker, P.: Miners trick stablecoin protocol pegnet, turning 11 into almost 7m hoard. CoinDesk (2020), https://www.coindesk.com/miners-trick-stablecoin-protocol-pegnet-turning-11-into-almost-7m-hoard

[13] Balancer Labs: BAL – balancer governance token (2020), https://docs.balancer.finance/protocol/bal-balancer-governance-token, accessed: 20-08-2020.

[14] Bano, S., Sonnino, A., Al-Bassam, M., Azouvi, S., McCorry, P., Meiklejohn, S., Danezis, G.: Sok: Consensus in the age of blockchains. In: Proceedings of the 1st ACM Conference on Advances in Financial Technologies. pp. 183–198 (2019)

[15] Bartoletti, M., Chiang, J.H.y., Lluch-Lafuente, A.: Sok: Lending pools in decentralized finance. arXiv preprint arXiv:2012.13230 (2020)

[16] Beck, R., Müller-Bloch, C., King, J.L.: Governance in the blockchain economy: A framework and research agenda. Journal of the Association for Information Systems **19**(10), 1 (2018)

[17] Beneš, N.: Introducing the dutchx (2017), https://blog.gnosis.pm/introducing-the-gnosis-dutch-exchange-53bd3d51f9b2

[18] Benhamouda, F., Halevi, S., Halevi, T.: Supporting private data on hyperledger fabric with secure multiparty computation. IBM Journal of Research and Development **63**(2/3), 3–1 (2019)

[19] bertcmiller: Tweet (2 April 2022), https://twitter.com/bertcmiller/status/1510249220967739398?t=Cf2PvmdsWyraKHNqOzYhwQ&s=19

[20] Biais, B., Bisiere, C., Bouvard, M., Casamatta, C.: The blockchain folk theorem. The Review of Financial Studies **32**(5), 1662–1715 (2019)

[21] Bitcoin, W.: Wbtc wrapped bitcoin an erc20 token backed 1:1 with bitcoin (2020), https://wbtc.network/

[22] BitMEX: Bitmex perpetual contracts guide (2020), https://www.bitmex.com/app/perpetualContractsGuide

[23] Blocknative: Evidence of mempool manipulation on black thursday: Hammerbots, mempool compression, and spontaneous stuck transactions (2020), https://www.blocknative.com/blog/mempool-forensics

[24] Bloomberg: How $60 Billion in Terra Coins Went Up in Algorithmic Smoke. https://www.bloomberg.com/graphics/2022-crypto-luna-terra-stablecoin-explainer/ (20 May 2022)

[25] Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W.: Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. In: 2015 IEEE symposium on security and privacy. pp. 104–121. IEEE (2015)

[26] Breidenbach, L., Daian, P., Tramèr, F., Juels, A.: Enter the hydra: Towards principled bug bounties and exploit-resistant smart contracts. In: 27th {USENIX} Security Symposium ({USENIX} Security 18). pp. 1335–1352 (2018)

[27] Buterin, V.: A next-generation smart contract and decentralized application platform. white paper **3**(37) (2014)

[28] bZx Network: bZx, The most powerful open finance protocol (2020), https://bzx.network/

[29] Carlsten, M., Kalodner, H., Weinberg, S.M., Narayanan, A.: On the instability of bitcoin without the block reward. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 154–167 (2016)

[30] CertiK: Yam finance smart contract bug analysis & future prevention (2020), https://certik.io/blog/technology/yam-finance-smart-contract-bug-analysis-future-prevention

[31] Chen, X., Park, D., Roşu, G.: A language-independent approach to smart contract verification. In: International Symposium on Leveraging Applications of Formal Methods. pp. 405–413. Springer (2018)

[32] Claburn, T.: Single-line software bug causes fledgling yam cryptocurrency to implode just two days after launch (2020), https://www.theregister.com/2020/08/13/yam_cryptocurrency_bug_governance/

[33] Clark, J.: The replicating portfolio of a constant product market. Available at SSRN 3550601 (2020)

[34] Coinbase: Coinbase (2020), https://www.coinbase.com/

[35] Cointelegraph: Compound liquidator makes $4m as oracles post inflated dai price (2020), https://cointelegraph.com/news/compound-liquidator-makes-4m-as-oracles-post-inflated-dai-price

[36] Compound: Compound finance (2019), https://compound.finance/

[37] Compound: Open price feed (2020), https://compound.finance/prices, accessed: 06-12-2020.

[38] ConsenSys: Mythril (2021), https://github.com/ConsenSys/mythril

[39] Consensys: Mythx: Smart contract security service for ethereum (2021), https://mythx.io/

[40] Cooper, T.: imbtc uniswap pool drained for ~$300k in eth (2020), https://defirate.com/imbtc-uniswap-hack/, accessed: 20-01-2021.

[41] Cousaert, S., Xu, J., Matsui, T.: Sok: Yield aggregators in defi. arXiv preprint arXiv:2105.13891 (2021)

[42] Cronje, A.: yEARN (2020), https://yearn.finance

[43] CryptoCompare: Cryptocompare exchange review, march 2022 (2022), https://www.cryptocompare.com/media/40124872/cryptocompare_exchange_review_2022_03_vf2.pdf

[44] Curve Finance: Curve.fi (2020), https://www.curve.fi/, accessed: 20-08-2020.

[45] Dafflon, J., Baylina, J., Shababi, T.: Eip-777: Erc777 token standard (2017), https://eips.ethereum.org/EIPS/eip-777

[46] Daian, P., Goldfeder, S., Kell, T., Li, Y., Zhao, X., Bentov, I., Breidenbach, L., Juels, A.: Flash boys 2.0: Frontrunning, transaction reordering, and consensus instability in decentralized exchanges. arXiv preprint arXiv:1904.05234 (2019)

[47] DeFi Pulse: What is defi? (2019), https://defipulse.com/blog/what-is-defi/

[48] DeFi Pulse: The decentralized finance leaderboard at defi pulse (2020), https://defipulse.com/

[49] Defiant: Iron Finance Implodes After 'Bank Run'. https://thedefiant.io/iron-finance-implodes-after-bank-run (17 June 2021)

[50] Defiant, T.: Bsc's venus protocol left with bad debt after liquidations (May 20, 2021), https://thedefiant.io/bscs-venus-protocol-left-with-bad-debt-after-liquidations/

[51] dForce: dforce (2020), https://dforce.network/

[52] Dubovitskaya, A., Ackerer, D., Xu, J.: A game-theoretic analysis of cross-ledger swaps with packetized payments (2021)

[53] dYdX: dydx (2019), https://dydx.exchange/

[54] Egorov, M.: Stableswap - efficient mechanism for stablecoin liquidity (2019), https://www.curve.fi/stableswap-paper.pdf

[55] Eskandari, S., Moosavi, S., Clark, J.: Sok: Transparent dishonesty: front-running attacks on blockchain. In: International Conference on Financial Cryptography and Data Security. pp. 170–189. Springer (2019)

[56] ETH Tx Decoder: Transaction analysis (2020), https://ethtx.info/mainnet/0x9d093325272701d63fdafb0af2d89c7e23eaf18be1a51c580d9bce89987a2dc1, accessed: 13-01-2021.

[57] Evans, A.: Liquidity provider returns in geometric mean markets. arXiv preprint arXiv:2006.08806 (2020)

[58] Feist, J.: Slither – a solidity static analysis framework (2018), https://blog.trailofbits.com/2018/10/19/slither-a-solidity-static-analysis-framework/

[59] Feng, F., Weickmann, B.: Set: A protocol for baskets of tokenized assets (2019), https://www.setprotocol.com/pdf/set_protocol_whitepaper.pdf

[60] Finance, B.: Tweet (14 February 2022), https://twitter.com/finance_build/status/1493223330685591558

[61] Flashbots: Flashbots Docs: Understanding Bundles. https://docs.flashbots.net/flashbots-auction/searchers/advanced/understanding-bundles (2022)

[62] Foundation, E.: Solidity v0.8.0 documentation (2020), https://docs.soliditylang.org/en/v0.8.0/index.html, accessed: 12-01-2020.

[63] Foxley, W.: $10.8m stolen, developers implicated in alleged smart contract 'rug pull'. CoinDesk (2020), https://www.coindesk.com/compounder-developers-implicated-alleged-smart-contract-rug-pull

[64] Foxley, W., De, N.: Weekend attack drains decentralized protocol dforce of $25m in crypto. CoinDesk (2020), https://www.coindesk.com/attacker-drains-decentralized-protocol-dforce-of-25m-in-weekend-attack

[65] Foxley, W.: Exploit during ethdenver reveals experimental nature of decentralized finance. CoinDesk (2020), https://www.coindesk.com/exploit-during-ethdenver-reveals-experimental-nature-of-decentralized-finance

[66] Frangella, E.: Crypto black thursday: The good, the bad, and the ugly. https://medium.com/aave/crypto-black-thursday-the-good-the-bad-and-the-ugly-7f2acebf2b83 (2020), accessed: 20-01-2021.

[67] Gnosis: API3 IDO incident - post mortem (2020), https://hackmd.io/@n6YCqowrQduQ5u25wSoRXw/Hylnk7SjD

[68] Gnosis: Introduction to gnosis protocol (2020), https://docs.gnosis.io/protocol/docs/introduction1/

[69] Godbole, O.: Defi flippening comes to exchanges as uniswap topples coinbase in trading volume. CoinDesk (2020), https://www.coindesk.com/defi-flippening-uniswap-topples-coinbase-trading-volume

[70] Gudgeon, L., Perez, D., Harz, D., Livshits, B., Gervais, A.: The decentralized financial crisis. In: 2020 Crypto Valley Conference on Blockchain Technology (CVCBT). pp. 1–15 (2020)

[71] Gudgeon, L., Moreno-Sanchez, P., Roos, S., McCorry, P., Gervais, A.: Sok: Off the chain transactions. IACR Cryptol. ePrint Arch. **2019**, 360 (2019)

[72] Gudgeon, L., Werner, S.M., Perez, D., Knottenbelt, W.J.: Defi protocols for loanable funds: Interest rates, liquidity and market efficiency. In: Proceedings of the 2nd ACM Conference on Advances in Financial Technologies. p. 92–112 (2020)

[73] Hanson, R.: Combinatorial information market design. Information Systems Frontiers **5**(1), 107–119 (2003)

[74] Harper, C.: Defi degens hit hard by eminence exploit will be partially compensated. CoinDesk (2020), https://www.coindesk.com/eminence-exploit-defi-compensated

[75] Harvest Finance: Harvest flashloan economic attack post-mortem (2020), https://medium.com/harvest-finance/harvest-flashloan-economic-attack-post-mortem-3cf900d65217, accessed: 29-12-2020.

[76] Harz, D., Gudgeon, L., Gervais, A., Knottenbelt, W.J.: Balance: Dynamic adjustment of cryptocurrency deposits. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 1485–1502 (2019)

[77] Harz, D., Knottenbelt, W.: Towards safer smart contracts: A survey of languages and verification methods. arXiv preprint arXiv:1809.09805 (2018)

[78] Hull, J., et al.: Options, futures and other derivatives/John C. Hull. Upper Saddle River, NJ: Prentice Hall, (2009)

[79] Huo, L., Klages-Mundt, A., Minca, A., Munter, F., Wind, M.: Decentralized Governance of Stablecoins with Closed Form Valuation. In *Mathematical Research for Blockchain Economy*. https://arxiv.org/abs/2109.08939 (2022)

[80] IDEX: Idex 2.0: The next generation ofnon-custodial trading. URL: https://idex.io/document/IDEX-2-0-Whitepaper-2019-10-31.pdf (2019)

[81] Index: Index: A comprehensive list of decentralized exchanges (dex)., https://distribuyed.github.io/index/

[82] Jones, S.P., Eber, J.M., Seward, J.: Composing contracts: an adventure in financial engineering. ACM SIG-PLAN Notices **35**(9), 280–292 (2000)

[83] Judmayer, A., Stifter, N., Zamyatin, A., Tsabary, I., Eyal, I., Gazi, P., Meiklejohn, S., Weippl, E.: Pay to win: Cheap, crowdfundable, cross-chain algorithmic incentive manipulation attacks on pow cryptocurrencies. Cryptology ePrint Archive, Report 2019/775 (2019), https://eprint.iacr.org/2019/775

[84] Kaleem, M., Shi, W.: Demystifying pythia: A survey of chainlink oracles usage on ethereum. arXiv preprint arXiv:2101.06781 (2021)

[85] Kalodner, H.A., Carlsten, M., Ellenbogen, P., Bonneau, J., Narayanan, A.: An empirical study of namecoin and lessons for decentralized namespace design. In: WEIS. Citeseer (2015)

[86] Kalra, S., Goel, S., Dhawan, M., Sharma, S.: ZEUS: analyzing safety of smart contracts. In: 25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018. The Internet Society (2018), http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_09-1_Kalra_paper.pdf

[87] Kao, H.T., Chitra, T., Chiang, R., Morrow, J.: An analysis of the market risk to participants in the compound protocol. In: Third International Symposium on Foundations and Applications of Blockchains (2020)

[88] Khatri, Y.: Dai price increase led to a massive $88 million worth of liquidations at defi protocol compound (2020), https://www.theblockcrypto.com/post/85850/dai-compound-dydx-liquidations-defi, accessed: 14-01-2021.

[89] Klages-Mundt, A.: Vulnerabilities in maker: oracle-governance attacks, attack daos, and (de)centralization (Nov 14, 2019), https://link.medium.com/VZG64fhmr6

[90] Klages-Mundt, A., Harz, D., Gudgeon, L., Liu, J.Y., Minca, A.: Stablecoins 2.0: Economic foundations and risk-based models. In: Proceedings of the 2nd ACM Conference on Advances in Financial Technologies. pp. 59–79 (2020)

[91] Klages-Mundt, A., Minca, A.: (in) stability for the blockchain: Deleveraging spirals and stablecoin attacks. Cryptoeconomic Systems (2021)

[92] Klages-Mundt, A., Minca, A.: While stability lasts: A stochastic model of non-custodial stablecoins. Mathematical Finance (2022)

[93] Koeppelmann, M.: Tweet (18 July 2020), https://twitter.com/koeppelmann/status/1284502534208528385

[94] Lee, B.E., Moroz, D.J., Parkes, D.C.: The political economy of blockchain governance. Available at SSRN 3537314 (2020)

[95] Lee, J.: Nubits (2014), https://nubits.com/NuWhitepaper.pdf

[96] Lee, L., Klages-Mundt, A.: Governance extractable value (Apr 23, 2021), https://ournetwork.substack.com/p/our-network-deep-dive-2

[97] Leshner, R., Hayes, G.: Compound: The money market protocol (2019), https://compound.finance/documents/Compound.Whitepaper.pdf

[98] Limited, T.: Tether: Fiat currencies on the bitcoin blockchain (2016), https://tether.to/wp-content/uploads/2016/06/TetherWhitePaper.pdf, accessed: 08-06-2020

[99] Lin, L.X., Budish, E., Cong, L.W., He, Z., Bergquist, J.H., Panesir, M.S., Kelly, J., Lauer, M., Prinster, R., Zhang, S., et al.: Deconstructing decentralized exchanges. Stanford Journal of Blockchain Law & Policy (2019)

[100] Liu, B., Szalachowski, P.: A first look into defi oracles (2020)

[101] Liu, M.: Urgent: Ousd was hacked and there has been a loss of funds (2020), https://medium.com/originprotocol/urgent-ousd-has-hacked-and-there-has-been-a-loss-of-funds-7b8c4a7d534c, accessed: 29-12-2020.

[102] LongForWisdom: [urgent] flash loans and securing the maker protocol (2020), https://forum.makerdao.com/t/urgent-flash-loans-and-securing-the-maker-protocol/490

[103] Luu, L., Chu, D.H., Olickel, H., Saxena, P., Hobor, A.: Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. pp. 254–269 (2016)

[104] Maker: The maker protocol: Makerdao's multi-collateral dai (mcd) system, https://makerdao.com/en/whitepaper/, accessed: 08-06-2020

[105] MakerDAO: Makerdao (2019), https://makerdao.com/en/

[106] Martinelli, F., Mushegian, N.: Balancer whitepaper: A non-custodial portfolio manager, liquidity provider, and price sensor. (2019), https://balancer.finance/whitepaper/, accessed: 26-08-2020.

[107] McCorry, P., Hicks, A., Meiklejohn, S.: Smart contracts for bribing miners. In: International Conference on Financial Cryptography and Data Security. pp. 3–18. Springer (2018)

[108] Nadler, M., Schär, F.: Decentralized finance, centralized ownership? an iterative mapping process to measure protocol token distribution. arXiv preprint arXiv:2012.09306 (2020)

[109] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)

[110] Narayanan, A., Bonneau, J., Felten, E., Miller, A., Goldfeder, S.: Bitcoin and cryptocurrency technologies: a comprehensive introduction. Princeton University Press (2016)

[111] Niemerg, A., Robinson, D., Livnev, L.: Yieldspace. https://yield.is/YieldSpace.pdf (2020)

[112] OpenCollective: cadcad (2020), https://cadcad.org/

[113] Opyn: Opyn (2020), https://opyn.co/#/

[114] opyn: Opyn eth put exploit (2020), https://medium.com/opyn/opyn-eth-put-exploit-c5565c528ad2

[115] Panja, S., Roy, B.K.: A secure end-to-end verifiable e-voting system using zero knowledge based blockchain. IACR Cryptol. ePrint Arch. **2018**, 466 (2018)

[116] PeckShield: 88mph incident: Root cause analysis (2020), https://peckshield.medium.com/88mph-incident-root-cause-analysis-ce477e00a74d

[117] PeckShield: bzx hack full disclosure (with detailed profit analysis) (2020), https://medium.com/@peckshield/bzx-hack-full-disclosure-with-detailed-profit-analysis-e6b1fa9b18fc

[118] Peckshield: Value defi incident: Root cause analysis (2020), https://peckshield.medium.com/value-defi-incident-root-cause-analysis-fbab71faf373, accessed: 13-01-2021.

[119] Percent Finance: Important announcement (2020), https://percent-finance.medium.com/important-announcement-d35f9a0df112

[120] Perez, D., Livshits, B.: Smart contract vulnerabilities: Does anyone care? arXiv preprint arXiv:1902.06710 (2019)

[121] Perez, D., Livshits, B.: Broken metre: Attacking resource metering in EVM. In: 27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020. The Internet Society (2020), https://www.ndss-symposium.org/ndss-paper/broken-metre-attacking-resource-metering-in-evm/

[122] Perez, D., Werner, S.M., Xu, J., Livshits, B.: Liquidations: Defi on a knife-edge. arXiv preprint arXiv:2009.13235 (2020)

[123] Perez, D., Xu, J., Livshits, B.: Revisiting transactional statistics of high-scalability blockchains. p. 535–550. IMC '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3419394.3423628, https://doi.org/10.1145/3419394.3423628

[124] Permenev, A., Dimitrov, D., Tsankov, P., Drachsler-Cohen, D., Vechev, M.: Verx: Safety verification of smart contracts. In: 2020 IEEE Symposium on Security and Privacy, SP. pp. 18–20 (2020)

[125] Peterson, J., Krug, J.: Augur: a decentralized, open-source platform for prediction markets. arXiv preprint arXiv:1501.01042 (2015)

[126] Pirus, B.: Cheese bank's multi-million-dollar hack explained by security firm (2020), https://cointelegraph.com/news/cheese-bank-s-multi-million-dollar-hack-explained-by-security-firm, accessed: 29-12-2020.

[127] Qin, K., Zhou, L., Livshits, B., Gervais, A.: Attacking the defi ecosystem with flash loans for fun and profit (2020)

[128] Raman, R.K., Vaculin, R., Hind, M., Remy, S.L., Pissadaki, E.K., Bore, N.K., Daneshvar, R., Srivastava, B., Varshney, K.R.: Trusted multi-party computation and verifiable simulations: A scalable blockchain approach. arXiv preprint arXiv:1809.08438 (2018)

[129] Rate, D.: Cream finance partially delists ftt amidst governance contention (2021), https://defirate.com/cream-ftt-delisting/

[130] Reijers, W., O'Brolcháin, F., Haynes, P.: Governance in blockchain technologies & social contract theories. Ledger **1**, 134–151 (2016)

[131] Rekt: Harvest finance - rekt (2020), https://rekt.ghost.io/harvest-finance-rekt/

[132] Rekt: Warp finance - rekt (2020), https://rekt.eth.link/warp-finance-rekt/

[133] Rekt: The big combo (growth defi - rekt) (2021), https://rekt.eth.link/the-big-combo/

[134] Rekt: Dodo - rekt (2021), https://rekt.eth.link/au-dodo-rekt/

[135] Rekt: Meerkat finance - bsc - rekt (2021), https://rekt.eth.link/meerkat-finance-bsc-rekt/

[136] Rekt: Paid network - rekt (2021), https://rekt.eth.link/paid-rekt/

[137] Rekt: Yearn - rekt (2021), https://rekt.eth.link/yearn-rekt/

[138] Reynolds, K., Pan, D.: Cover protocol attack perpetrated by 'white hat,' funds returned, hacker claims. CoinDesk (2020), https://www.coindesk.com/cover-protocol-attack-perpetrated-by-white-hat-all-funds-returned-hacker-claims

[139] Robinson, D.: Etherum is a dark forest (2020), https://medium.com/@danrobinson/ethereum-is-a-dark-forest-ecc5f0505dff, accessed: 24-11-2020.

[140] Rodler, M., Li, W., Karame, G.O., Davi, L.: Sereum: Protecting existing smart contracts against re-entrancy attacks. In: Proceedings of 26th Annual Network & Distributed System Security Symposium (NDSS) (February 2019), http://tubiblio.ulb.tu-darmstadt.de/111410/

[141] Roughgarden, T.: Algorithmic game theory. Communications of the ACM **53**(7), 78–86 (2010)

[142] Roughgarden, T.: Transaction fee mechanism design for the ethereum blockchain: An economic analysis of eip-1559. arXiv preprint arXiv:2012.00854 (2020)

[143] samczsun: Escaping the dark forest (2020), https://samczsun.com/escaping-the-dark-forest, accessed: 24-11-2020.

[144] Schär, F.: Decentralized finance: On blockchain-and smart contract-based financial markets. FRB of St. Louis Review (2021)

[145] Swende, M.: Blockchain frontrunning (2017), https://swende.se/blog/Frontrunning.html

[146] Synthetix: Litepaper (2020), https://docs.synthetix.io/litepaper/, accessed: 06-12-2020

[147] Synthetix: Synthetix | decentralised synthetic assets (2020), https://www.synthetix.io

[148] Tarasov, A.: Millions lost: The top 19 defi cryptocurrency hacks of 2020 (2020), https://cryptobriefing.com/50-million-lost-the-top-19-defi-cryptocurrency-hacks-2020/

[149] Thompson, P.: Defi project pickle finance exploited for $20 million (2020), https://coingeek.com/defi-project-pickle-finance-exploited-for-20-million/

[150] Thurman, A.: Value defi protocol suffers $6 million flash loan exploit (2020), https://cointelegraph.com/news/value-defi-protocol-suffers-6-million-flash-loan-exploit, accessed: 29-12-2020.

[151] Tokenlon: imbtc (2020), https://tokenlon.im/imBTC#/

[152] Tolmach, P., Li, Y., Lin, S.W., Liu, Y.: Formal analysis of composable defi protocols. arXiv preprint arXiv:2103.00540 (2021)

[153] Tornado: Tornado (2021), https://tornado.cash/

[154] Torres, C.F., Schütte, J., State, R.: Osiris: Hunting for integer bugs in ethereum smart contracts. In: Proceedings of the 34th Annual Computer Security Applications Conference. p. 664–676. ACSAC '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3274694.3274737, https://doi.org/10.1145/3274694.3274737

[155] Tsankov, P., Dan, A., Drachsler-Cohen, D., Gervais, A., Buenzli, F., Vechev, M.: Securify: Practical security analysis of smart contracts. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 67–82 (2018)

[156] Uniswap: Uniswap (2020), https://app.uniswap.org/#/swap

[157] Uniswap: Uniswap whitepaper (2020), https://hackmd.io/@HaydenAdams/HJ9jLsfTz#%F0%9F%A6%84-Uniswap-Whitepaper, accessed: 26-08-2020.

[158] von Wachter, V., Jensen, J.R., Ross, O.: Measuring asset composability as a proxy for ecosystem integration. arXiv preprint arXiv:2102.04227 (2021)

[159] Wallet, W.: Wasabi wallet (2021), https://wasabiwallet.io/

[160] Wang, Y., Kogan, A.: Designing confidentiality-preserving blockchain-based transaction processing systems. International Journal of Accounting Information Systems **30**, 1–18 (2018)

[161] Warren, W., Bandeali, A.: 0x: An open protocol for decentralized exchange on the ethereum blockchain. URL: https://github.com/0xProject/whitepaper (2017)

[162] Werner, S.M., Pritz, P.J., Perez, D.: Step on the gas? A better approach for recommending the ethereum gas price. arXiv preprint arXiv:2003.03479 (2020)

[163] Wintermute, M.: Hegic: On-chain options trading protocol on ethereum powered by hedge contracts and liquidity pools (2020), https://ipfs.io/ipfs/QmWy8x6vEunH4gD2gWT4Bt4bBwWX2KAEUov46tCLvMRcME, accessed: 13-11-2020.

[164] Winzer, F., Herd, B., Faust, S.: Temporary censorship attacks in the presence of rational miners. In: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). pp. 357–366. IEEE (2019)

[165] Wood, G., et al.: Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper **151**(2014), 1–32 (2014)

[166] Wright, T.: Akropolis defi protocol 'paused' as hackers get away with $2m in dai (2020), https://cointelegraph.com/news/akropolis-defi-protocol-paused-as-hackers-get-away-with-2m-in-dai, accessed: 29-12-2020.

[167] Xu, J., Ackerer, D., Dubovitskaya, A.: A game-theoretic analysis of cross-chain atomic swaps with htlcs (2020)

[168] YAM: Yam finance (2020), https://yam.finance/

[169] YAM Finance: Yam post-rescue attempt update (2020), https://medium.com/@yamfinance/yam-post-rescue-attempt-update-c9c90c05953f

[170] yearn: Incident disclosure 2021-10-27. https://github.com/yearn/yearn-security/blob/master/disclosures/2021-10-27.md (Oct 27, 2021)

[171] Zamyatin, A., Al-Bassam, M., Zindros, D., Kokoris-Kogias, E., Moreno-Sanchez, P., Kiayias, A., Knottenbelt, W.J.: Sok: communication across distributed ledgers. IACR Cryptol. ePrint Arch. (2020)

[172] Zamyatin, A., Harz, D., Lind, J., Panayiotou, P., Gervais, A., Knottenbelt, W.: Xclaim: Trustless, interoperable, cryptocurrency-backed assets. In: 2019 IEEE Symposium on Security and Privacy (SP). pp. 193–210. IEEE (2019)

[173] Zcash: Zcash (2021), https://z.cash/

[174] Zhang, F., Cecchetti, E., Croman, K., Juels, A., Shi, E.: Town crier: An authenticated data feed for smart contracts. In: Proceedings of the 2016 aCM sIGSAC conference on computer and communications security. pp. 270–282 (2016)

[175] Zhang, R., Xue, R., Liu, L.: Security and privacy on blockchain. ACM Computing Surveys (CSUR) **52**(3), 1–34 (2019)

[176] Zhang, Y., Chen, X., Park, D.: Formal specification of constant product (xy= k) market maker model and implementation (2018), https://github.com/runtimeverification/verified-smart-contracts/blob/uniswap/uniswap/x-y-k.pdf

[177] Zhao, W., Li, H., Yuan, Y.: Understand volatility of algorithmic stablecoin: Modeling, verification and empirical analysis. arXiv preprint arXiv:2101.08423 (2021)

[178] Zhou, L., Qin, K., Cully, A., Livshits, B., Gervais, A.: On the just-in-time discovery of profit-generating transactions in defi protocols. arXiv preprint arXiv:2103.02228 (2021)

[179] Zhou, L., Qin, K., Torres, C.F., Le, D.V., Gervais, A.: High-frequency trading on decentralized on-chain exchanges. arXiv preprint arXiv:2009.14021 (2020)

A DEFI PROTOCOLS

Table 2: DeFi Protocols: A selection of prominent DeFi protocols classified according to the proposed protocol types.

Exchanges	PLFs	Stablecoins	Portfolio Managers	Derivatives
Curve	Compound	Maker	Harvest	Opyn
Uniswap	Aave	Unit	Yearn	Hegic
Sushiswap	dYdX	Reflexer	Set	Synthetix
Balancer	Cream	Fei	Alpha	
Bancor				
1inch				

B EMPIRICAL EXPLOITS

There have been a range of exploits in DeFi applications. This is a non-exhaustive list of some of the exploits and vulnerabilities referenced in Sections 5 and 6.

Reentrancy Exploits.

- *dForce:* One of the most prominent examples of this exploit was against the dForce protocol [51], which features a PLF, in April 2020 to drain around 25 million USD worth of funds [64].The attacker used imBTC [151], which is an ERC-777 token [45], to perform the attack. A particularity of ERC-777 tokens, as opposed to ERC-20 tokens, is that they have a hook calling the receiver when the receiver receives funds. This means that any ERC-777 tokens will indirectly result in the receiver having control of the execution. In the dForce attack, the attacker used this reentrancy pattern to repeatedly increase their ability to borrow without enough collateral to back up their borrow position, effectively draining the protocol's funds.
- *imBTC Uniswap Pool:* Despite the fact that Uniswap does not support ERC-777 tokens [157], an imBTC Uniswap [156] pool worth roughly 300 000 USD was drained using the reentrancy attack.

Integer Manipulation Exploits.

- *YAM:* In August 2020, the YAM protocol [168], which had locked almost 500 million USD worth of tokens in a very short period of time, realized that there was an arithmetic-related bug.Two integers scaled to their base unit were multiplied and the result not scaled back, making the result orders of magnitude too large [30, 32].This prevented the governance to reach quorum and locked all the funds in the protocol's treasury contract, effectively locking over 750 000 USD worth of tokens [169] indefinitely.

Logical Bug Exploits.

- *bZx:* In September 2020, the bZx protocol [28], a lending protocol, suffered a loss of over 8 million USD due to a trivial logic error [117], despite having been through two independent audits. The bZx protocol uses its own ERC-20 tokens, which are minted by locking collateral and repaid to redeem the locked collateral. As other ERC-20 tokens, bZx tokens allow users to transfer the tokens. However, due to a logical bug, when a user transferred tokens to themselves, the amount transferred would effectively only be added to their balance, and not correctly subtract from it, allowing a user to double his amount of tokens at will. The tokens created could then be used to withdraw funds that the attacker never owned or locked.

Market Manipulation.

- *Venus:* Since pre-printing this paper, a clear exploit that manipulated this market structure was performed on the Venus protocol [50]. In this exploit, the attacker manipulated the thinly traded XVS market and borrowed large amounts of BTC against XVS collateral at the manipulated high prices. This led to $100m of bad debt (effectively, the profit for the attacker) in the protocol when the XVS market equilibrated to normal pricing.

C BATCH SETTLEMENT SYSTEMS

In Gnosis exchange [68], trades are matched algorithmically in periodic batches maintained by decentralized keepers. Keepers compete to solve a complicated matching problem. They submit solutions on-chain, from which the protocol executes the best solution, by some metric. If this keeper market is competitive, trades should be settled at fair prices, though issues can arise when the keeper market is not competitive [93] or if the method for choosing the best keeper solution can be gamed [67].