Market-based Security for Distributed Applications

George Bissias[†] Brian N. Levine[†] Nikunj Kapadia[*]
[†]College of Information and Computer Sciences
[*]Isenberg School of Management
University of Massachusetts Amherst, MA, USA

ABSTRACT

Ethereum contracts can be designed to function as fully decentralized applications called DAPPs that hold financial assets, and many have already been fielded. Unfortunately, DAPPs can be hacked, and the assets they control can be stolen. A recent attack on an Ethereum decentralized application called The DAO demonstrated that smart contract bugs are more than an academic concern. Ether worth hundreds of millions of US dollars was extracted by an attacker from The DAO, sending the value of its tokens and the overall exchange price of ether itself tumbling.

We present two market-based techniques for insuring the ether holdings of a DAPP. These mechanisms exist and are managed as part of the core programming of the DAPP, rather than as separate mechanisms managed by users. Our first technique is based on futures contracts indexed by the trade price of ether for DAPP tokens. Under fairly general circumstances, our technique is capable of recovering the majority of ether lost from theft with high probability even when all of the ether holdings are stolen; and the only cost to DAPP token holders is an adjustable ether withdrawal fee. As a second, complementary, technique we propose the use of *Gated Public Offerings (GPO)* as a mechanism that mitigates the effects of attackers that exploit DAPP withdrawal vulnerabilities. We show that using more than one public offering round encourages attackers to exploit the vulnerability early, or depending on certain factors, to delay exploitation (possibly indefinitely) and short tokens in the market instead. In both cases, less ether is ultimately stolen from the DAPP, and in the later case, some of the losses are transferred to the market.

CCS CONCEPTS

• **Security and privacy** → ; **Economics of security and privacy**; *Software security engineering*;

KEYWORDS

Smart contracts, software security, financial derivatives, game theory

ACM Reference format:
George Bissias, Brian N. Levine, and Nikunj Kapadia. 2017. Market-based Security for Distributed Applications. In *Proceedings of ACM New Security Paradigms Workshop, Santa Cruz, CA USA, Oct 2017 (NSPW'17),* 16 pages.
https://doi.org/10.1145/3171533.3171541

1 INTRODUCTION

Ethereum [24] is a blockchain-based currency [32] designed to serve as a general purpose computing system with Turing-complete smart contracts [7]. Ethereum's contracts can be designed to function as fully *decentralized applications* called *DAPPs*. Many DAPPs have already been fielded, including an online marketplace [6], a role playing game [2], a prediction market [3], and an Internet service provider [5].

Unfortunately, DAPPs can be hacked, and the assets they control can be stolen. For example, in May 2016, a DAPP called the *Decentralized Autonomous Organization* (The DAO) was created as a type of decentralized hedge fund. It raised over US$150,000,000 worth of *ether*, Ethereum's native cryptocurrency, during a crowd sale in which *tokens* were issued in tranches at a fixed exchange rate for ether [34]. The ether was stored in and controlled by the software. Token holders were allowed to submit proposals for how to invest The DAO's ether, and tokens also afforded the holders with voting rights on what proposals should be pursued. For example, one proposal was for investing the funds in a particular startup. A key feature of The DAO was that ether could be withdrawn. Token holders were allowed to split off a *child-DAO* at any time, taking with them their share of the ether. After a provisional 28-day waiting period, the child-DAO would be allowed to execute proposals itself, which could be to withdraw its ether holdings.

On June 17, 2016 an attacker began an unauthorized transfer of ether from The DAO into a new child-DAO [21]. By June 18, more than US$100,000,000 in ether had been stolen, locked in the child-DAO for the 28-day waiting period, after which the attacker could take possession [34]. The attack was not due to a flaw or vulnerability in Ethereum; rather it was a flaw in the DAO's programming. The Ethereum community struggled over how to handle the theft. Ultimately, the community was literally divided when the majority of the Ethereum developers decided to rewrite the blockchain in order to return the ether to the original token holders, while a smaller portion continued to mine on the old blockchain [20]. Even on the old blockchain, a *white hat* group of hackers were able to steal back a large portion of the stolen ether [35]. Nevertheless, the hack led to the dissolution of The DAO, split the community, and caused the price of ether to drop by approximately two thirds.

New Opportunities. The DAO attacker took advantage of an overlooked flaw in the logic of the software contract. Since the demise of the The DAO, similar DAPPs have been launched (e.g., DigixDAO [9]). Presumably, these new DAOs have employed additional methods of software security, ranging from manual code audits to static analysis [16, 26] and formal verification [14, 19]. Unfortunately, DAPPs will inevitably increase in complexity, leading to new opportunities for stored ether to be withdrawn without authorization. Traditional computer science research will seek to advance techniques for securing this code to protect DAPP assets.

We propose that DAPPs offer opportunities for novel security mechanisms. Unlike other software, smart contracts supported by cryptographic currencies in particular have an intrinsic and precise economic value that can be quantified and insured. And in the case of The DAO and many DAPPS, the contract software literally stores ether that are accumulated during its *Initial Coin Offering* or ICO. Typically, the ether held in a DAPP is redeemable for its outstanding tokens. The ether can also be transferred (i.e., *invested*) in other contracts that offer goods, services, or collateral to the DAPP and its constituent token holders. A DAPP's main security objective should be to ensure that the withdrawal and transfer of its ether is possible only by redeeming the appropriate number of tokens.

For years, software has been used to secure the financial instruments that drive markets — the roles can be reversed as well. Indeed, the application of insurance markets to software security has been studied before [15]. In this paper, we propose that market-based security solutions can be coded into the core of the software itself and managed directly by it, rather than exist as a separate mechanism employed by consumers and businesses.

Contributions. In this paper, we present two complementary market-based techniques for securing assets held by distributed applications on systems like Ethereum. The first technique insures the value of ether holdings of a DAPP by employing futures contracts that are indexed by the trade price of ether for DAPP tokens. We are not proposing a method of actually securing contract code, but rather we have developed a process for protecting a DAPP's assets in the (perhaps inevitable) event that a software security flaw is discovered and exploited. Under fairly general circumstances, *our technique is capable of recovering the majority of ether lost from theft with high probability even when all of the ether holdings are stolen*; and the only cost to DAPP token holders is an adjustable ether withdrawal fee. The probability of successful recovery is tied to the volatility of the futures contracts. For example, suppose that it takes as many as d days for investors to become aware of a DAPP theft where all ether are stolen. If the probability of a margin call in d days is $1 - p$ for a futures contract with 20 times leverage, then our approach will allow for the recovery of half the stolen ether with probability p and a withdrawal fee of 5%. A higher withdrawal fee of 25% allows for more than 80% of the ether to be recovered with probability p.

As a second, complementary, technique we propose the use of *Gated Public Offerings* (GPO) as a mechanism for mitigating the effects of attackers that exploit DAPP withdrawal vulnerabilities. A GPO separates the public offering of tokens into multiple rounds. During the time period between the first and last public offering of tokens, attackers can choose to either exploit the vulnerability, taking all ether held in the contract, or they can short contract tokens on the market, which transfers the risk. We introduce a game-theoretical framework for modeling and manipulating the incentives of one or more attackers with knowledge of a vulnerability in a DAPP. We show that using more than one round of public offerings encourages attackers to exploit the vulnerability early, or depending on certain factors, to delay exploitation (possibly indefinitely) and short tokens in the market first. In the former case, less ether is stolen overall from the DAPP and fewer token holders are exposed to the theft. In the latter case less ether is ultimately stolen from the DAPP and some of the loss is transferred to the market.

2 PROBLEM STATEMENT

In this paper, we explore new paradigms for securing assets held by software. To that end, we seek answers to several fundamental questions.

Overall motivation:

> *How can we secure the financial assets held within a DAPP despite the presence of a vulnerability in the contract code whose exploitation could lead to the complete loss of those assets?*

Although it may be too difficult to absolutely determine the security of a contract's *software*, it still may be possible to secure the contract's *assets* by using financial derivatives as a form of insurance and leveraging market dynamics to influence attacker behavior. Therefore, we further seek to answer two secondary questions.

DAPP insurance:

> *Can we design a DAPP insurance mechanism based on common financial derivatives so that the DAPP can continue to allow ether withdrawals, yet it is also able to recover a significant portion of lost ether in the event of catastrophic theft?*

Disincentivizing theft:

> *Is there a way to alter incentives using market forces so as to coerce agents with knowledge of a vulnerability to either exploit the vulnerability before all tokens have been issued or otherwise avoid stealing all the available ether?*

We attempt to answer the questions above through the introduction of two distinct, but complementary, techniques. We develop DAPP insurance by imposing a configurable fee on all ether withdrawals. The fee is used to purchase short futures contracts that hedge against a potential drop in the trade price of tokens in the event that the withdrawal is later determined to be unauthorized. Our approach to disincentivizing theft is simple in contrast. We argue that by dividing the sale of tokens among rounds of a gated public offering, it is possible to encourage attackers to reveal a vulnerability without stealing all the available ether. Before proceeding, we pause to layout the assumptions that we make for the analysis in later sections.

Basic assumptions. We assume that DAPP $\mathcal{D}$ issues each token (TOK) in exchange for exactly 1 ether (ETH). Prior to attack, it is assumed that all n ETH remain in $\mathcal{D}$ and are available for withdrawal. The attacker is assumed to be capable of conducting an unauthorized withdrawal of m ETH without sacrificing any TOK. The DAO tokens traded on an open market and carried an exchange price denominated in ETH. Thus, we assume that TOK can also be traded for ETH. Throughout this document, we refer to the *exchange* *price* of ETH for TOK, denoted $\frac{\text{ETH}}{\text{TOK}}$ (ether per token). We further assume that one of the existing futures exchanges [1, 4] will implement *futures contracts* for speculating on the $\frac{\text{ETH}}{\text{TOK}}$ exchange price. We feel this is a reasonable assumption given that tokens are implemented as distinct cryptocurrencies and exchanges have been quick to implement futures contracts for popular cryptocurrencies. Finally, we assume that, like ether, tokens cannot be stolen due to flaws in cryptographic protections. Rather, only the contract code is assumed to be at risk, since it is code specific to $\mathcal{D}$.

Idle holdings. Although most DAPPs invest some portion of their ETH by moving assets out of the contract, we wish to isolate fluctuations in the price of $\frac{\text{ETH}}{\text{TOK}}$ in our analysis to those caused by the deposit or withdrawal of ETH in exchange for TOK. Thus we assume that $\mathcal{D}$ makes no investments with its ETH. The only time ETH is deposited is by means of a public offering of TOK and the only time ETH is removed is during a (possibly unauthorized) withdrawal.

DAPP price assumption. We also make one more assumption, which is critical to our approach: the exchange price $\frac{\text{ETH}}{\text{TOK}}$ is equal to the ratio of ETH held by $\mathcal{D}$ (less withdrawal fees) to the number of TOK outstanding. Specifically, prior to attack the price of $\frac{\text{ETH}}{\text{TOK}}$ is $1 - f$ and an unauthorized withdrawal of $m \leq n$ ETH will result in a drop to $(1 - f)(\frac{n-m}{n})$. This assumption is justified if we assume that the trade of price of $\frac{\text{ETH}}{\text{TOK}}$ is determined solely by the amount of ETH that can be redeemed for each TOK. Overall, it allows us to isolate the performance of our mechanisms in our analysis. In future work, it would be interesting to relax this assumption and incorporate other price influencers such as press coverage, market shocks, and price movement of other assets controlled by $\mathcal{D}$.

3 BACKGROUND AND RELATED WORK

Ethereum. Using a blockchain as a method for distributed consensus was first proposed by Nakamoto as part of a proposal for a cryptocurrency called Bitcoin [32]. Blockchains allow for an open group of peers to reach consensus through a *proof of work* mining algorithm, which mitigates Sybil attacks [22] and other limitations [25, 36].

Strictly for explanatory purposes, Ethereum [24] can be thought of as a system that starts from the mechanisms in Bitcoin but with native support for Turing-complete smart-contracts built in. Ethereum's mining process incentivizes miners to execute the code that comprises contracts by rewarding them with *ether*, a cryptocurrency that, like bitcoin, has an exchange rate with fiat currencies, such as the US dollar. In addition to regular accounts analogous to Bitcoin addresses, Ethereum implements *contract accounts*, which hold a balance of ether and whose behavior is controlled by arbitrarily complex code. This code can be triggered to execute when a user (or another contract) submits a transaction to the miners which specifies the contract to run. Transactions also provide the ether necessary to pay for computation.

Both Bitcoin and Ethereum support markets with voluminous trading, including futures markets. And the tokens issued by currently operational DAPPs are also available on several different cryptocurrency exchanges. Figure 1 shows the volume of Yunbi's spot market for the DigixDAO token (DGD) against the Yen (CNY) as an example [12]. Bitmex is also a popular market and it offers bitcoin swaps, bitcoin futures, and Ethereum futures [1].

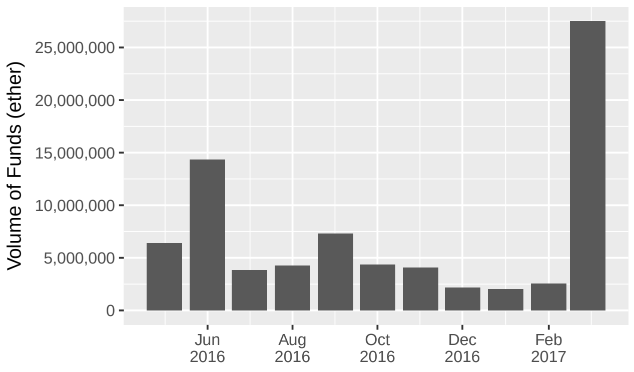


Figure 1: Volume of Digix DAO futures on the Yunbi market exchange, which represents about 73% of the futures market for Digix DAO. (source: coinmarketcap.com.)

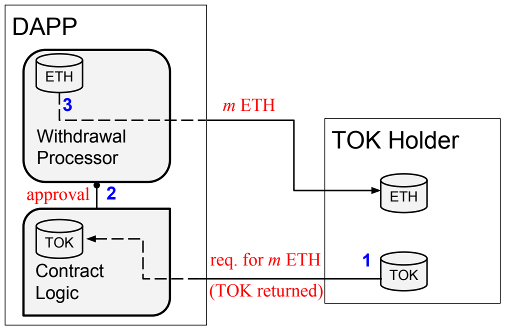


Figure 2: Typical mechanism for withdrawal of stored assets from today's DAPPs.

DAPP Withdrawals. The current process for withdrawals of assets from a DAPP $\mathcal{D}$ is illustrated in Figure 2, and works as follows.

(1) Token holders authorize an exchange of their TOK for ETH via a message to $\mathcal{D}$'s contract logic.
(2) The message's authenticity is confirmed;
(3) and the withdrawal processor returns ETH to the client.

In the case of The DAO heist, the withdrawal message was authentic, but a flaw in the contract logic allowed the attacker to, in the end, use a relatively small amount of TOK to withdraw more ETH than the TOK were worth.

Related work. Security, economics, and game theory have been intimately linked by researchers for more than a decade. One of the first works concerning the application of financial incentives to the problem of computer security comes from Camp and Wolfram [18], who propose economic sanctions for parties that fail to patch known vulnerabilities. Anderson and Moore [13] offer a slightly different perspective on economics and security, which advocates for the creation of vulnerability markets as well as exploit insurance. Example works that relate to this paper include an analysis of the role of market insurance among the players of a security game by Johnson et al. [28], and an evaluation of how incentives can shift actors towards protection or insurance by Grossklags et al. [27].

Böhme and Schwartz provide an excellent survey of the broad topic of market models of cyber insurance and risk transfer [15]. And Schwalb [33] explores political, legal, and economic aspects of the creation of a vulnerabilities market.

More recently, Manshaei et al. [30] provide a comprehensive overview of related works that have applied game theory to security. Egelman et al. [23] discusses the ethical implications of the formation and participation in vulnerabilities markets. Laszka and Grossklags analyze whether insurance providers should take a proactive role in improving software security [29]. Finally, Marotta et al. [31] offer a survey of cyber-insurance mechanisms.

Smart contract security. Research on smart-contract security is nascent in the cryptocurrency community. Buterin [17] offers some general recommendations based on the types of attacks witnessed in Ethereum to-date. He advocates for the development of layered, incremental defenses, but also argues that *identifying* an attack is fundamentally difficult. In particular, he notes that it is difficult to objectively and unanimously identify and mitigate DAPP hacks from the outside because members of the community have such different perspectives.

Tethered Currencies. Several cryptocurrencies address the problem of price stability by tethering to an asset that is stable. Techniques for managing market volatility are related to our goal in that they seek to protect asset value, but they do not protect against security vulnerabilities in contract code. For example, Tether is an appropriately named blockchain-based currency that is backed one-to-one with the US dollar [11]. Similarly, Digix DGX is an Ethereum-based DAPP that issues tokens tethered to ounces of actual gold [10]. As mentioned above, DixigDAO is a DAO that profits from Digix fees, and it has a separate token DGD that is untethered [9]. Finally, the Dai Credit System is not tethered to a fiat currency or commodity [8]. Instead, it is backed by other digital assets as collateral to stabilize its price.

4 A DAPP INSURANCE MECHANISM

A major problem with Ethereum contracts is that the logic can be very complex. Contract software can represent an ample opportunity for an attacker to exploit a vulnerability in the contract to conduct an unauthorized ETH withdraw. In this section, we introduce a mechanism that protects the ether held in a contract *even in the event that an attacker is capable of making an unauthorized withdrawal at no expense.* Our solution is for the contract to retain a withdrawal fee from the ether to be withdrawn that will be used to purchase insurance, in the form of futures contracts FUT that hedge against a decrease in the trade price of $\frac{\text{ETH}}{\text{TOK}}$. If the withdrawal was unauthorized, then presumably the trade price will eventually drop. At this point the contract closes its FUT in exchange for ETH, recovering some portion of the stolen ETH. However, in order to prevent iterative attacks, the recovered ETH are not immediately made available for withdrawal. Instead they are locked in a recovery cache until TOK holders vote for their release (after the withdrawal code has been patched).

Our approach is not meant as a replacement for methods that secure the execution of code or uncover vulnerabilities through program analysis. Instead, we intend for our mechanisms to complement such methods.

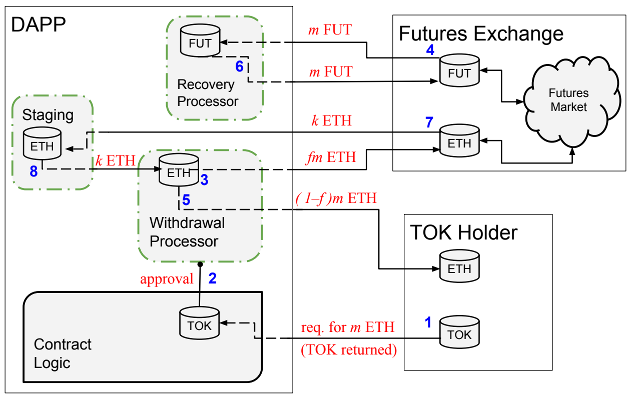


Figure 3: Our proposed mechanism for asset withdrawals. The Withdrawal Processor, Recovery Processor, and Staging mechanisms form a trusted security base for our Market-Secured Software approach.

4.1 Withdrawal Workflow

Figure 3 shows the modified withdrawal workflow. We have introduced three general purpose withdrawal components (shown in dash-dotted, green) that operate independently from each other and from the existing contract-specific logic. The idea is that the code used to implement these components can be thoroughly tested and reused among all DAPPs that implement a withdrawal mechanism. The withdrawal process unfolds in eight steps (each delineated with a blue number in the figure), which we describe in detail below for a contract $\mathcal{D}$ as defined in Section 2.

(1) An existing token holder interacts with the contract-specific logic of $\mathcal{D}$ in some way so as to initiate the withdrawal of m ETH. In general, withdrawal is due to the deposit of TOK, but more complex scenarios can also trigger a withdrawal.
(2) The contract logic signals its approval of the withdrawal of m ETH to the withdrawal processor.
(3) The withdrawal processor removes m ETH from its cache and separates a fraction fm for the withdrawal fee. The fee is passed to an account controlled by $\mathcal{D}$ on the futures exchange. The $(1-f)m$ remaining ETH are momentarily held by the withdrawal processor.
(4) The withdrawal processor trades fm ETH on the futures exchange for m futures contracts FUT that hedge against a drop in the trade price of $\frac{\text{ETH}}{\text{TOK}}$. The FUT are moved from an account controlled by $\mathcal{D}$ on the futures exchange to a cache controlled by the recovery processor.
(5) The withdrawal processor releases $(1 - f)m$ ETH to the user who initiated the withdrawal. This is the final responsibility of the withdrawal processor.
(6) The recovery processor monitors the value of the FUT held in the account controlled by $\mathcal{D}$ on the futures exchange. Eventually the recovery processor trades the FUT for ETH (say, k ETH).
(7) Upon withdrawal from the futures exchange, the ETH are stored in a staging area that is intentionally quarantined from the rest of the contract logic.

(8) Periodically, the current TOK holders vote to release the ETH in the staging area back into the main ETH cache controlled by the withdrawal processor. This manual voting process is used to ensure that an attacker cannot drain the ETH holdings of $\mathcal{D}$ multiple times before a potential exploit is discovered.

It is possible that the FUT will be liquidated (lost) by way of margin call before the exploit occurs due to a rise in price of $\frac{\text{ETH}}{\text{TOK}}$, which would render useless any protection from those FUT. We will address that complication in Section 6 as part of more detailed analysis of this mechanism.

4.2 Trusted Core Software

As should be clear, to employ the financial mechanisms above, we cannot escape the need to secure at least some portion of the DAPP's software using traditional methods. In particular, our approach relies on a software-only trusted core, which is comprised of the withdrawal processor, recovery processor, and staging region. Importantly, the remaining contract logic, which controls other mechanisms specific to that DAPP are not part of the core. Other contract logic can include receiving and sending of messages from other accounts and contracts, overseeing voting on policies suggested by token holders (excluding the vote to release ETH from staging), receiving tokens, checking if funds are sufficient for enacting policies, contacting services external to Ethereum, and more. Each of these other functions represent a potential avenue for some type of attack; for example, an attacker might bring a policy to vote without being a token holder. However, the focus of our core is strictly on the withdrawal of ETH from the DAPP (the dash-dotted, green elements in Figure 3).

Our expectation is that once programmed, the core can be re-used in many DAPPs as a key component, just as cryptographic implementations are heavily inspected and re-used rather than reimplemented each time. We leave the implementation of this trusted core for future work.

5 MECHANICS OF FUTURES CONTRACTS

Futures contracts, or simply *futures*, are instruments that are created and brokered by a *futures exchange.* They are an agreement between two parties, which are issued in pairs to individuals who enter opposing *positions.* Central to the mechanics of futures contracts is the notion of the *index price*, or the exchange price of $\frac{\text{ETH}}{\text{TOK}}$ on some well-known currency exchange (typically independent from the futures exchange). Each contract is stipulated in terms of the *settlement price*, which is the average index price during the hour prior to the *settlement date.* We next define a particular type of futures contract that hedges against fluctuations in the exchange price of $\frac{\text{ETH}}{\text{TOK}}$.

5.1 Contracts and Markets

The financial instruments we describe here and employ in Section 6 are based on common mechanisms implemented by multiple cryptocurrency exchanges [1, 4].

DEFINITION 1: A *futures contract* is an agreement between *guarantor* and *beneficiary* whereby the guarantor agrees to pay the beneficiary 1 TOK worth of ETH at the settlement price on the settlement date. A trader *opens* a *long position* (LP) or *short position* (SP) whenever she becomes the beneficiary or guarantor, respectively, of a futures contract.

A long position affords a trader with the *right* to receive ETH at settlement while a short position encumbers a trader with the *obligation* to deliver ETH at settlement. New futures are initially created directly by the exchange which matches pairs of traders willing to assume opposing positions.

Markets. An open position is automatically closed at settlement, but it is also possible for a trader to close a position by transferring her rights or obligations to a new trader. This transfer is conducted on the *futures market.* In this market, a trader buys a futures contract in order to open a long position, and he sells the contract in order to close a long position; alternatively, a trader sells to open a short position, and buys a contract to close a short position. "Buying" is understood to mean either becoming the beneficiary of a contract in the case of LPs; or being released from the obligation of a contract in the case of SPs. Similarly, "selling" when holding an LP means giving up the right as beneficiary of the contract while it means becoming guarantor of a contract when entering an SP. The trade price, or just *price*, refers the amount of ETH a trader agrees to pay when buying a contract; alternatively, it's the amount received when selling. When the futures market is *efficient*, futures will trade close to the current index price indicating that any information about the future settlement price is captured by the current index price.

Profit. Profits and losses associated with holding a position on a futures contract closely track the movement of the underlying index price.

THEOREM 1: Assuming efficient markets, a *fall* in price from $\varepsilon\ \frac{\text{ETH}}{\text{TOK}}$ to $(1-p)\varepsilon\ \frac{\text{ETH}}{\text{TOK}}$ while holding a short position will result in an opposing *profit* of $p\varepsilon$ ETH at the settlement date. Similarly, a rise to $(1+p)\varepsilon\ \frac{\text{ETH}}{\text{TOK}}$ will result in a *loss* of $p\varepsilon$ ETH.

PROOF: Suppose that an SP is opened such that a trader receives ε ETH in exchange for encumbering himself as the guarantor of a futures contract. Assuming that the market is efficient, the trade price would have been $\varepsilon\ \frac{\text{ETH}}{\text{TOK}}$. If the price subsequently falls to $(1-p)\varepsilon\ \frac{\text{ETH}}{\text{TOK}}$, then the guarantor is obligated to pay the beneficiary just $((1-p)\varepsilon)$ ETH at the time of settlement. Thus, the guarantor has earned profit $(\varepsilon - \varepsilon(1-p))$ ETH $= p\varepsilon$ ETH. Similarly, a rise in price results in a loss: $(\varepsilon - \varepsilon(1+p))$ ETH $= -p\varepsilon$ ETH. ■

An analogous theorem holds for LPs, where it can be shown, for example, that a drop to $(1-p)\varepsilon\ \frac{\text{ETH}}{\text{TOK}}$ will result in a *loss* of $p\varepsilon$ ETH. However, the focus of this paper is on the short positions opened by $\mathcal{D}$.

Notice that SPs have an unbounded potential downside and a finite potential upside of 100% gain, which occurs when the index price of $\frac{\text{ETH}}{\text{TOK}}$ drops to zero.

Normally there are fees associated with futures contracts. However, they are typically relatively small, and for clarity we omit them as part of our analysis.

5.2 Buying on Margin

Margin and Leverage. We have shown that holding a position can result in either a profit or loss as a contract's trade price fluctuates. Any losses are payable when the position is closed or upon contract settlement. For each open position, a trader maintains *margin* with the exchange, which is ETH collected at the time the position is opened for the express purpose of *covering* any potential losses. In order to enforce the futures contract, the exchange ensures that every trader has enough margin to cover each open position in the event that it was closed at the current market price. When a trader lacks sufficient margin to cover her open position, the exchange initiates a *margin call*: margin is confiscated by the exchange and used to close the open position. The trader loses both the margin and any rights or obligations associated with the futures contract.

A *λ-leveraged* futures contract requires only fraction $1/\lambda$ TOK worth of ETH in margin, where typically $1 \leq \lambda \leq 100$ in digital currency markets; e.g., see BitMEX [1].

DEFINITION 2: A λ-SP is a λ-leveraged short position that can be opened by posting $1/\lambda$ TOK worth of ETH in margin at the current trade price; alternatively, a λ-LP long position can be opened with $1/\lambda$ TOK worth of ETH.

Note that a trader can open several SPs at once. For readers that are unfamiliar with these standard mechanisms, we provide illustrative examples in Appendix A.

6 ANALYSIS OF THE INSURANCE MECHANISM

In this section, we provide technical details for the withdrawal and recovery processors, and derive results that prove the minimum ETH that can be recovered after theft under various assumptions. We introduce a theoretical framework for reasoning about the tradeoff between cost and security and provide concrete guidance on how to set security parameters.

At a high level, our approach is to charge a withdrawal fee and use that fee to purchase λ-SPs to hedge against a drop in the trade price of $\frac{\text{ETH}}{\text{TOK}}$. In general, more ETH can be recovered after theft when either a higher fee is collected during withdrawal or higher leverage is used. Thus using higher leverage is attractive because it allows for strong recovery without raising fees. However, using high leverage also raises the probability of margin call prior to the theft, which results in lost λ-SPs and prevents recovery. $\mathcal{D}$ must also decide *when* to close the λ-SPs so as to simultaneously hedge against losses due to theft and margin call. We find that a simple price threshold can be used to balance the tradeoff between these two competing objectives.

6.1 Withdrawal Processor

Suppose initially that $\mathcal{D}$ holds n ETH and that n TOK are outstanding. $\mathcal{D}$ will implement a particular withdrawal mechanism such that any time m ETH are withdrawn, it will reserve $(m\frac{\delta}{\lambda})$ ETH in order to open δm λ-SP short positions, as defined in Section 5. We say that $\mathcal{D}$ has implemented a (δ, λ) *insurance policy* and requires a $m\frac{\delta}{\lambda}$ *fee*. Because of the fee imposed, the n ETH stored in $\mathcal{D}$ are worth

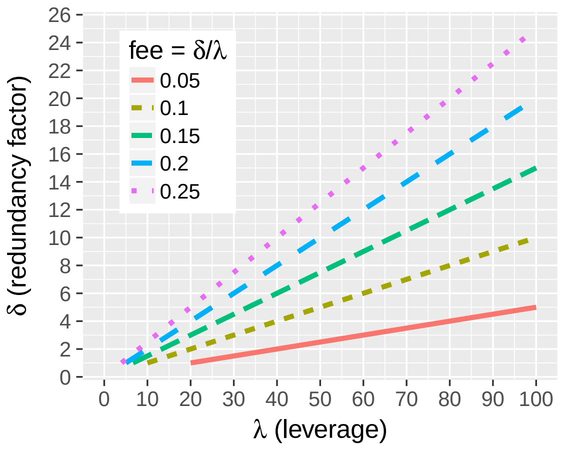


Figure 4: Valid values of leverage $1 \leq \lambda \leq 100$ and redundancy $\delta \geq 1$, such that fee $f = \delta/\lambda$.

$(1 - \frac{\delta}{\lambda})$ each when removed; i.e., $f = \frac{\delta}{\lambda}$ in terms of the fee variable we introduced in Section 4.

Figure 4 shows the tradeoff between λ and δ with each line representing a fixed withdrawal fee. For example, in order to maintain a 5% withdrawal fee (red line), the leverage can be as low as $\lambda = 1/f = 20$. On the other hand, to maintain the same fee when $\delta = 5$ we must set leverage to its highest value at $\lambda = 100$.

Now imagine that the m ether were withdrawn illegitimately as a result of an attacker exploiting some vulnerability in $\mathcal{D}$ — in other words, the attacker is capable of completing the withdrawal without depositing any TOK. After the withdrawal, $\mathcal{D}$ holds $n - m$ ETH and there are n TOK outstanding. Thus, by the DAPP Price Assumption, the exchange price will drop, but not all the way to $(1 - f)(n - m)/n$ $\frac{\text{ETH}}{\text{TOK}}$ each because the λ-SPs still hold value, a fact that investors will be aware of. $\mathcal{D}$ will then close the λ-SPs, reclaiming some amount of the stolen ETH, during a *recovery process.*

THEOREM 2: If there are n TOK outstanding and $\mathcal{D}$ implements a (δ, λ) insurance policy, then when m ETH are illegitimately withdrawn the quantity of ETH held by $\mathcal{D}$ after recovery will be equivalent to

$$\frac{\delta m + (n - m)}{1 + \frac{\delta m}{n}(1 - f)} \text{ETH}. \tag{1}$$

PROOF: According to Theorem 1, absent any exogenous influences, each λ-SP will increase in value by p in the event that the price of $\frac{\text{ETH}}{\text{TOK}}$ drops by p according to Theorem 1. Assuming that the withdrawal is seen as being unauthorized, investors will consider m ETH to have been stolen. On the other hand, $\mathcal{D}$ still retains δm λ-SPs as well as $(n - m)$ ETH. We seek to determine $\mathcal{E}(m, n, \delta, \lambda)$, the total ETH remaining in $\mathcal{D}$ after recovery. Let $\Pi(m, n, \delta, \lambda)$ be the profit of each SP when closed. The two functions are related by the following system of two equations. First, since $\mathcal{D}$ is holding $n - m$ ETH and δm SPs that are worth $\Pi(\cdot)$ each, we have:

$$\mathcal{E}(m, n, \delta, \lambda) = \quad n - m + \delta m\, \Pi(m, n, \delta, \lambda), \tag{2}$$

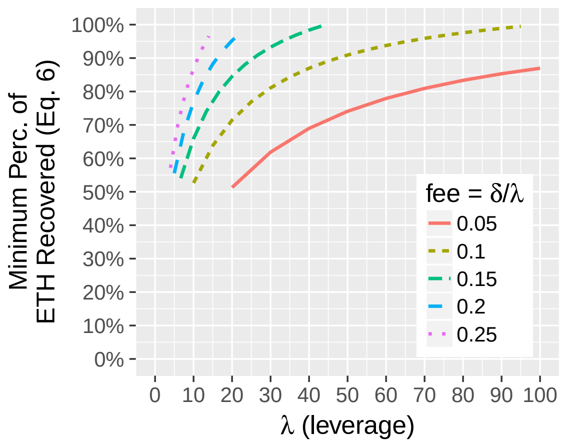


Figure 5: An illustration of Theorem 3 for $n = 1$ and various values of $\delta \geq 1$ and $1 \leq \lambda \leq 100$, such that $\lambda > \delta^2$. Note that without our proposed mechanism, the percentage of recovered stolen ETH would be 0.

Second, $\mathcal{D}$'s assets after recovery are $\mathcal{E}(\cdot)$, which when withdrawn are reduced to $1 - f$ of their value. Thus, the price of $\frac{\text{ETH}}{\text{TOK}}$ for the n remaining tokens will drop. The resulting ETH value of the SPs is the complement,

$$\Pi(m, n, \delta, \lambda) = \quad 1 - \frac{(1 - f)\mathcal{E}(m, n, \delta, \lambda)}{n}, \tag{3}$$

which follows from the DAPP Price Assumption and Theorem 1.

Solving these equations gives

$$\mathcal{E}(m, n, \delta, \lambda) = \frac{\delta m + (n - m)}{1 + \frac{\delta m}{n}(1 - f)} \tag{4}$$

$$\Pi(m, n, \delta, \lambda) = \frac{\lambda n + (\delta - \lambda)(n - m)}{\lambda n - \delta m(\delta - \lambda)} \tag{5}$$

where Eq. 4 holds the final result. ■

Example. Suppose that all n ETH were illegitimately withdrawn from the contract. If $\delta = 1$ and $\lambda = 20$, then according to Eq. 4, $\mathcal{D}$ can restore the ETH holdings to $\frac{20n}{39}$, which is nearly half of the total ether. Keeping $\lambda = 20$ and increasing δ to 2 recovers $\frac{40n}{56}$, which is about $\frac{2}{3}$ of the ETH. Note that in these scenarios we have recovered a significant portion of ETH even after *all* of the ETH initially in $\mathcal{D}$ was stolen (except for the fee f). Any ETH recovered was from closing the λ-SPs that $\mathcal{D}$ acquired during the withdrawal process.

This example raises a question: for fixed λ and δ, what is the least ETH that will remain in $\mathcal{D}$ after recovery for any amount of m stolen ETH?

THEOREM 3: If there are n TOK outstanding and $\mathcal{D}$ implements a (δ, λ) insurance policy with $\lambda > \delta^2$, then the least amount of ETH remaining after recovery is

$$\frac{n}{1 + \frac{1}{\delta} - \frac{\delta}{\lambda}} \tag{6}$$

regardless of how much ETH is initially stolen.

PROOF: We begin by finding the derivative of $\mathcal{E}(m, n, \delta, \lambda)$ with respect to m,

$$\frac{\partial \mathcal{E}}{\partial m}(m, n, \delta, \lambda) = \frac{\lambda n^2(\delta^2 - \lambda)}{(\lambda n + \delta\lambda m - \delta^2 m)^2}. \tag{7}$$

When $\lambda > \delta^2$, $\frac{\partial \mathcal{E}}{\partial m}$ is negative and therefore $\mathcal{E}$ is a decreasing function of m. It follows that $\mathcal{E}$ achieves its minimum value when $m = n$. Substituting $m = n$ back into $\mathcal{E}$ we have the final result. ■

Example. For illustrative purposes, Figure 5 plots a sample of values that result from Theorem 3. When the withdrawal fee is $f = 15\%$ and the leverage is $\lambda = 20$, then $\delta = (0.15)20 = 3$, and our constraint that $\lambda > \delta^2$ is met. As a result, the minimum fraction of ETH recoverable is $\frac{60n}{71}$, i.e., about 85%. Token holders in $\mathcal{D}$ may also enjoy lower fees with the same recovery percentage; we see from Figure 5 that about the same minimum recovery of 83% can be had from a fee of just $f = 5\%$ if leverage is set to $\lambda = 80$ with a redundancy of $\delta = 4$. However, as we discuss in the remainder of this section, the choice of leverage parameter λ is critical — greater leverage increases the chance of a margin call from relatively smaller jumps in the $\frac{\text{ETH}}{\text{TOK}}$ exchange rate, which wipes out the insurance protection provided by SP contracts.

6.2 Recovery Processor

The last section discussed properties of the withdrawal processor, which is designed to help $\mathcal{D}$ hedge against the loss of ETH due to theft. Specifically, $f = \frac{\delta}{\lambda}$ of each withdrawn ETH should be retained for the purpose of opening δ λ-SPs. Although we argued that this practice allows for the recovery of a significant portion of stolen ETH, there remains the question of *when* the λ-SPs should be closed.

$\mathcal{D}$ is autonomous and up to this point, the withdrawal mechanism has also remained autonomous. Ideally $\mathcal{D}$ will also be capable of deciding when to sell λ-SPs without direct intervention from token holders. On the other hand, it is difficult (if not impossible) to automatically detect theft, even when using the trade price of $\frac{\text{ETH}}{\text{TOK}}$ as a signal. More significantly, open λ-SPs are a valuable but volatile asset. The longer a λ-SP remains open, the more likely it is to receive a margin call if the exchange price $\frac{\text{ETH}}{\text{TOK}}$ fluctuates by more than $\frac{1}{\lambda}$ before the positions are closed.

Consider a simple sale criterion for λ-SPs based only on the current value of open contracts. Assuming efficient markets, each λ-SP can rise in value by as much as 1 ETH if the index price drops to 0 (see Section 5). We evaluate the simple approach of selling each λ-SP during recovery once its value rises above *recovery threshold* α ETH. Note that in this analysis we assume the λ-SPs are closed at an equilibrium price that is *at or above* α, as opposed to the threshold value α exactly.

THEOREM 4: If there are n TOK outstanding and $\mathcal{D}$ implements a (δ, λ) insurance policy with $\lambda > \delta^2$ and recovery threshold α defined such that $\frac{\delta}{\lambda} \leq \alpha < \frac{1}{\delta+1}$, then there will remain at least

$$\min\left[\frac{n}{1 + \frac{1}{\delta} - \frac{\delta}{\lambda}},\; n\left(1 - \frac{\lambda\alpha - \delta}{(\lambda - \delta)(1 - \alpha\delta)}\right)\right] \text{ETH} \tag{8}$$

after an unauthorized withdrawal of any size (not exceeding n) and subsequent recovery (if it occurs).

PROOF: Theorem 2 provides the value of a λ-SP for the unauthorized withdrawal of m ETH assuming that the λ-SPs are sold and their ETH added back to $\mathcal{D}$:

$$\Pi(m, n, \delta, \lambda) = \frac{\lambda n + (\delta - \lambda)(n - m)}{\lambda n - \delta m(\delta - \lambda)}.$$

By construction, the λ-SPs are sold when $\Pi(m, n, \delta, \lambda) > \alpha$. Since $\delta \geq 1$ and $\lambda > \delta^2$, we know that $\lambda > \delta$. It is given that $\alpha < \frac{1}{\delta}$. Therefore $\Pi(m, n, \delta, \lambda) > \alpha$ whenever

$$m > \frac{n(\lambda\alpha - \delta)}{(\lambda - \delta)(1 - \alpha\delta)} = m^*. \quad (9)$$

Because we assume that $\lambda > \delta^2$ and $\frac{\delta}{\lambda} < \alpha < \frac{1}{\delta+1}$, all terms in the r.h.s. of the inequality are greater than or equal to zero, and the denominator is strictly non-zero. Thus it must be the case that $m^* \geq 0$.

Now we have two cases. First, if $m > m^*$ then the λ-SPs are sold, and the minimum ETH after recovery is given by Theorem 3:

$$\frac{n}{1 + \frac{1}{\delta} - \frac{\delta}{\lambda}}. \quad (10)$$

Second, if instead $m \leq m^*$, then no λ-SPs are sold; in that case, the ETH remaining in $\mathcal{D}$ is as little as

$$n - m^* = n\left(1 - \frac{\lambda\alpha - \delta}{(\lambda - \delta)(1 - \alpha\delta)}\right). \quad (11)$$

Since m can be arbitrary, we choose the minimum ETH that can remain for either the case where $m > m^*$ or $m \leq m^*$. The result follows. ■

Example 1. To illustrate how the choice of α affects the amount of ETH recovered, suppose that $\delta = 1$, $\lambda = 20$, and we choose $\alpha = \frac{1}{2}$ for $\mathcal{D}$. Then Equation 11 is equal to $\frac{n}{19}$, while Equation 10 is equal to $\frac{20n}{39}$. Therefore, according to Theorem 4, with a large enough theft, as little as $\frac{n}{19}$ ETH might remain in $\mathcal{D}$ after recovery.

Example 2. However, if we decrease $\mathcal{D}$'s recovery threshold to $\alpha = \frac{1}{20}$, then Equation 11 increases to n. In this case Equation 8 is dominated by Equation 10 (still equal to $\frac{20n}{39}$). Thus Theorem 4 predicts in this case that more than half the original ETH will remain after recovery, no matter how much ETH was stolen.

Example 3. If we set $\alpha = \frac{\delta}{\lambda}$, its lowest possible value, then Equation 11 is equal to n for any valid values of δ and λ. In that case Equation 10 dominates the bound in Theorem 4 making it equivalent to the bound in Theorem 3.

Indeed, as long as we choose α such that

$$\frac{\delta}{\lambda} \leq \alpha \leq \frac{\lambda^2}{\delta^4 4 - \delta^3\lambda - 2\delta^2\lambda + 2\delta\lambda^2 + \lambda^2},$$

the minimum recoverable ETH (for any size m) will always be the same as in Theorem 3: $\frac{n}{1+\frac{1}{\delta}-\frac{\delta}{\lambda}}$. Thus the *worst case* amount of ETH remaining after recovery will be the same for any choice of α in that interval.

Nevertheless, there are substantive differences between choices of α.

- For lower values of α, recovery will be triggered earlier, which implies a lower risk of margin call. But because recovery is triggered earlier, less ETH will be recovered since the λ-SPs will be worth less.

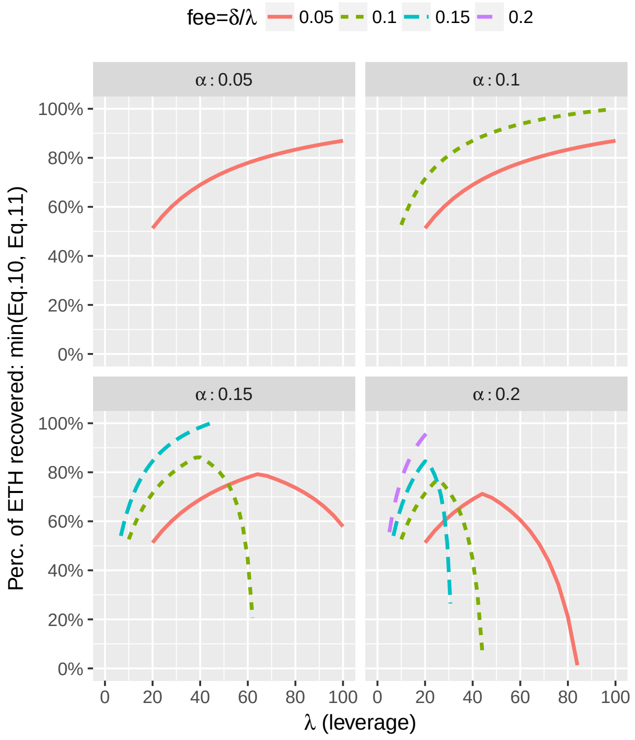


Figure 6: The percentage of stolen ETH recovered when $\mathcal{D}$ automatically closes its λ-SPs when their value is greater than threshold α. (Because we require that $\frac{\delta}{\lambda} \leq \alpha < \frac{1}{\delta+1}$, not all fees are applicable to each selection of α and λ.) In contrast, without our proposed mechanism, the percentage of recovered stolen ETH would be 0.

- On the other hand, a higher choice of α will trigger recover later, which means a higher risk of margin call but more ETH recovered.

6.3 Setting Security Parameters

Two parameters controlled by $\mathcal{D}$ are critical to the security of the DAPP withdrawal mechanism:

- λ, the SP leverage; and
- δ, the number of λ-SPs purchased for each ETH withdrawn.

Setting these parameters presents a tradeoff in terms of security and usability. The risk of margin call decreases with λ, however it also raises the withdrawal fee. And the minimum ETH recoverable goes up with δ, but it too raises the withdrawal fee.

It is difficult to predict the probability of a margin call for λ-SPs without knowing the dynamics of the futures market or the volatility of the underlying index price of $\frac{\text{ETH}}{\text{TOK}}$. To get a better idea of the price dynamics of a real cryptocurrency, we analyzed Bitcoin futures data from the OkCoin exchange from Feb 6, 2016 until April 2, 2017. For various values of λ, Figure 7 shows the empirical probability that a λ-SP will receive a margin call after a given period of time. After 3 days, the probability of a margin call is less than

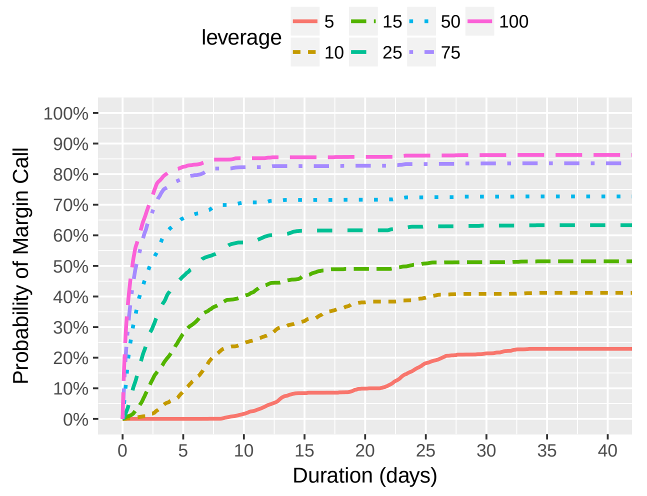


Figure 7: Probability of margin call based on real market data from Bitcoin futures data. (OkCoin exchange, Feb 6, 2016 – April 2, 2017).

40% when $\lambda = 25$, but when $\lambda = 100$ it becomes greater than 80%. Bitcoin is the oldest and most popular cryptocurrency, thus we expect that its price volatility will be lower than a newly introduced DAPP token. Accordingly, the estimated probabilities of margin call given in Figure 7 should be interpreted as lower bounds on the corresponding probabilities for TOK.

6.4 Contracts Without Exploits

It is also worth thinking about what happens to the ETH holdings of $\mathcal{D}$ and the trade price of $\frac{\text{ETH}}{\text{TOK}}$ on the spot market in the event that no vulnerability is ever exploited. If the price of TOK relative to ETH in the futures market is nevertheless highly volatile, then it is likely that the λ-SPs purchased by the withdrawal processor will receive a margin call before $\mathcal{D}$ can close them. In that case, according to the DAPP price assumption, the spot price should remain $(1 - f)\frac{\text{ETH}}{\text{TOK}}$.

However, if the futures market is relatively stable, and α is selected so that the λ-SPs are always closed before margin call, then the ETH collected in fees will eventually be returned to the ETH store of the withdrawal processor. If a total of m ETH have been withdrawn, this will raise the number of ETH in $\mathcal{D}$ to $n - (1 - f)m$ when there are $n - m$ TOK outstanding. Hence, when $m > \frac{n}{(2-f)}$, the DAPP price assumption dictates that the value of $\frac{\text{ETH}}{\text{TOK}}$ will exceed 1, even with the withdrawal fee. This means that investors that hold their TOK will receive a premium on their initial investment equivalent to a portion of the withdrawal fees paid by prior TOK holders that withdrew their ETH early.

6.5 Limitations

There are several practical limitations that affect the techniques presented in this section. First, we have made some simplifying assumptions about the nature of the spot and futures market for TOK. To the extent that any of these assumptions are untrue, our analysis could be misleading or incorrect. For example, critical to our approach is the assumption of a liquid futures market that enjoys relatively low volatility. But if the TOK market turns out to be highly volatile, then $\mathcal{D}$ will be required to charge exorbitant fees, which could slow adoption or destroy the DAPP entirely.

Second, our solution does nothing to actually prevent theft. Economists like to say that there is no such thing as a free lunch, a truism that holds for market-secured software. When an attacker successfully exploits a vulnerability in $\mathcal{D}$, stealing m ETH, someone must pay for that theft, even if $\mathcal{D}$ itself is capable of recovering much of the lost ETH. Our approach offloads the burden of the theft to the market; the parties that pay are the traders who purchased λ-LPs opposing the λ-SPs purchased by $\mathcal{D}$. A valid question is whether it is preferable for market participants to be punished instead of DAPP investors.

7 DISINCENTIVIZING DAPP THEFT

The DAPP insurance mechanism introduced in Section 4 is capable of recovering a significant fraction of ETH stolen by means of exploitation of a software vulnerability; but it does nothing to alter an attackers behavior. In this section, we show that by publicly offering TOK incrementally, $\mathcal{D}$ can substantially mitigate losses due to theft. Because of the uncertainty over when a competitor might exploit the vulnerability, attackers have incentive both the exploit early and to short TOK to ensure that they benefit from a theft. These competing goals conspire to leave less ETH in $\mathcal{D}$ at the time of exploitation.

Unless it is possible to take measures to slow the rate that a vulnerability can be exploited, a single public offering will inevitably leave all ETH exposed to attack. That is to say, there is no way to prevent attackers from stealing all ETH raised by $\mathcal{D}$. In contrast, if even a single additional pubic offering is introduced, it is always possible to encourage attackers to exploit the vulnerability early or delay exploitation (possibly indefinitely) and short TOK in the market instead. In the former case, less ETH is stolen overall from $\mathcal{D}$ and fewer TOK holders are exposed to the theft. The latter case is less desirable for TOK holders because the value of TOK will drop to zero when the vulnerability is revealed. However, the shorted TOK actually encourage arbitragers to withdraw ETH from $\mathcal{D}$, which means that less ETH is ultimately stolen from $\mathcal{D}$ and instead the loss is transferred to the market.

Gated Public Offerings In the previous sections, we have assumed that all outstanding TOK are issued by DAPP $\mathcal{D}$ in an initial public offering that trades n TOK for n ETH. Thus, as soon as a vulnerability is discovered, an attacker is immediately capable of exploiting it for maximal profit. Alternatively, $\mathcal{D}$ can conduct a *gated* public offering whereby TOK are gradually introduced to the market. For example, $\mathcal{D}$ could offer n new tokens every 30 days during a year-long period. In this case, the attacker faces a dilemma: if he exploits right away, he misses out on future ETH deposits, but if he waits too long to exploit the vulnerability, then another attacker might strike first.

In fact, the attacker has a third option: she can sell TOK short in the spot market, knowing that the trade price of $\frac{\text{ETH}}{\text{TOK}}$ will drop once the vulnerability is exploited. This approach has two key benefits for the attacker: *(i)* more ETH will flow into the contract before she exploits; and *(ii)* the attacker is certain to profit from the attack even if she is not the one to perform the exploit.

Overall fewer TOK holders are affected by a theft if attackers exploit the vulnerability right away. But directing would-be attackers toward shorting TOK is actually good behavior for the community in the sense that it tends to lower the price of $\frac{\text{ETH}}{\text{TOK}}$, which naturally encourages arbitragers to drain ETH from $\mathcal{D}$ before an attack occurs.

7.1 Assumptions and Preliminaries

For ease of exposition, in this section we put aside the insurance mechanism introduced earlier as well as the notion of a withdrawal fee. Note however that gated public offerings and the insurance mechanism introduced in Section 4 are entirely compatible. As before, we continue to assume that an attacker with knowledge of a vulnerability is capable of stealing all ETH from $\mathcal{D}$ at the cost of no TOK. And any attacker with knowledge of a vulnerability is assumed to have gained this knowledge prior to the first public offering from $\mathcal{D}$. If the attacker discovers a flaw after the completion of all public offerings, then this approach cannot be used; in that case, $\mathcal{D}$ must rely exclusively on the insurance mechanism from Section 4. Finally, if a futures market exists, we assume that the market depth is no greater than the depth of the spot market. We explain the reason for this assumption in Appendix B; it is not fundamental, but simplifies our analysis.

To evaluate the efficacy of using a gated public offering, we must first investigate more deeply the mechanism behind the DAPP Price Assumption (Section 2). In the absence of fees, the DAPP Price Assumption states that the trade price of $\frac{\text{ETH}}{\text{TOK}}$ will remain equivalent to the ratio of ETH holdings to TOK outstanding, which we will call the *equilibrium price*. Consider how this equilibrium is reached under normal market conditions. In particular, suppose that a potential attacker, with knowledge of a vulnerability, decides to short 1 TOK by borrowing it from an existing TOK holder and immediately selling it on the spot market (we assume no interest is charged for borrowing). In general, we can expect that the price of $\frac{\text{ETH}}{\text{TOK}}$ will drop somewhat because of the increase in supply. At that point, an arbitrageur will purchase the TOK at a discount and immediately redeem it for 1 ETH, making a small profit in the process.

There is no way to know for sure what the price of $\frac{\text{ETH}}{\text{TOK}}$ will be after the arbitrageur makes her purchase, but in this section we assume that the price will return to equilibrium. We refer to this assumption as the *symmetry of supply and demand* in the market. We feel that the assumption is reasonable and, although not fundamental to our approach, it does allow for simpler analysis. However in future work it would be interesting to explore how incentives change during public offerings when this assumption is relaxed. For example, as more and more TOK are redeemed by arbitragers, a market scarcity arises. And to the extent that the DAPP is valued beyond its ETH holdings (a relaxation of the DAPP price assumption), the trade price of TOK could actually rebound beyond the equilibrium price. This dynamic would fundamentally change the game that we analyze below.

7.2 The DAPP Heist Game

We wish to demonstrate the improved utility for $\mathcal{D}$ in increasing the number of public offerings of TOK. To that end, we construct the *DAPP heist game*, which is conducted between two attackers A_1 and A_2, each having knowledge of a vulnerability that allows them to steal all the ETH in $\mathcal{D}$. Our focus is on the two-player game assuming either a single or double round public offering only. We feel that concentrating on these games is sufficient to demonstrate the value of a gated public offering. However it is straightforward to extend our analysis to more players and additional rounds.

Game definition. The game unfolds over a maximum of k public offerings that we call *rounds*, but will end prematurely once one of the attackers exploits the vulnerability. At the beginning of round i, DAPP $\mathcal{D}$ offers n_i TOK for sale to the public in exchange for 1 ETH each. Just after the public offering in round i, attacker A_j decides if she will immediately exploit the vulnerability or instead short some amount $m_{i,j}$ of the newly offered TOK. If multiple attackers attempt to exploit the vulnerability in the same round, then we assume that they split the ETH in the contract equally. Likewise, if multiple attackers attempt to short all the available TOK, then we assume that they are able to short equal amounts. It is possible that no attacker ever exploits the vulnerability, but regardless we assume that the vulnerability is eventually exposed so that the TOK price drops to 0 and attackers profit maximally from the TOK they shorted.

In a given round, we assume that the set of all attackers can short fraction $0 \leq \gamma \leq 1$ of the TOK introduced that round before any other attackers can successfully exploit the vulnerability. This assumption allows us to model either a disparity in the time it takes to short TOK versus exploiting the vulnerability or a handicap in value (if for example there is a significant expense associated with shorting TOK). As an example, we can set $\gamma > \frac{1}{2}$ to model the case where $\mathcal{D}$ imposes a restrictive daily ETH withdrawal limit, which might slow down an exploit. Or alternatively, set $\gamma \ll \frac{1}{2}$, to be conservative in assuming that an exploit can be carried out rapidly. The value γ is a prominent factor in our analysis.

Strategies. Before the game commences, each attacker develops a master strategy of shorting TOK for zero or more rounds, and then optionally exploiting the vulnerability in the subsequent round. Specifically, attacker A_j plans a strategy of the form $S_{1,j}, S_{2,j}, \ldots, E_{i,j}$ over $i \geq 1$ rounds; where $S_{r,j}$ denotes shorting $m_{r,j}$ TOK in round r, and $E_{i,j}$ denotes exploiting the vulnerability in final round i. To be clear, an attacker can optionally exploit immediately and never short; and similarly, an attacker can short tokens but never exploit. Note that strategy $S_{i,j}$ is a *meta strategy* that itself represents an entire set of strategies parameterized by the number of TOK $m_{i,j}$ to be shorted.

For convenience, we define $N_i = \sum_{t=1}^{i} n_t$ and $M_{i,j} = \sum_{t=1}^{i} m_{t,j}$. Note that because it is not possible to short more TOK than the quantity that have been offered, it follows that $M_{i,j} \leq N_i$ for all i.

Strategy sets. A *strategy set* for the DAPP heist game is a vector comprised of a single strategy chosen by each player. The *payoff* for a strategy set is a vector with the potential profit for each attacker given the chosen strategies. For a two-player game, payoffs for all strategy sets can be represented by a *payoff matrix*. Table 1 depicts the payoff matrix for a game where two attackers compete for the

		$\mathbf{A_2}$	
		$S_{1,2}$	$E_{1,2}$
$\mathbf{A_1}$	$S_{1,1}$	$(m_{1,1}; m_{1,2})$	$(\gamma m_{1,1}; n_1 - \gamma m_{1,1})$
	$E_{1,1}$	$(n_1 - \gamma m_{1,2}; \gamma m_{1,2})$	$(\frac{n_1}{2}; \frac{n_1}{2})$

Table 1: Payoff matrix for the 2-attacker, single-round public offering DAPP heist game. Because the matrix is symmetric, we gray out the upper diagonal for clarity. When $\gamma > \frac{1}{2}$ only $(S_{1,1}; S_{1,2})$ is the Nash equilibrium. If $\gamma < \frac{1}{2}$, then $(E_{1,1}; E_{1,2})$ is the Nash equilibrium. If $\gamma = \frac{1}{2}$, then every meta strategy is a Nash equilibrium. In other words, only when $\mathcal{D}$ strongly limits the daily ETH withdrawals (i.e., $\gamma > \frac{1}{2}$) it is both attackers' best strategy to short rather than exploit immediately.

ETH holdings of $\mathcal{D}$ over the course of a single public offering. A Nash equilibrium for the game occurs for any strategy set where no change in strategy for player A_i, can result in a higher payoff if the strategies for other players are held fixed.

Evaluating meta strategies. As mentioned above, the strategy $S_{i,j}$ is a meta strategy that encompasses all strategies where A_j shorts $m_{i,j} \leq n_i$ TOK in round i. Thus associated with $S_{i,j}$ is a separate strategy for each value of $m_{i,j}$. Strategy sets that incorporate one or more meta strategies constitute *meta strategy sets* or M-sets; each forms a sub-game with its own Nash equilibrium.

In all the games we consider, every M-set has exactly one Nash equilibrium, which we call an *extremal strategy set*, or E-set. And we label its corresponding payoff the *extremal payoff* for the M-set. For example, $(S_{1,1}; S_{1,2})$ is an M-set for the single round game where attackers A_1 and A_2 short $m_{1,1}$ and $m_{1,2}$ TOK respectively (neither ever exploits). Because both attackers would like to short as much TOK as possible, but only n_1 were released in the public offering, the E-set corresponding to that M-set is for each to short $\frac{n_1}{2}$ TOK. Therefore, the extremal payoff vector is given by $(\frac{n_1}{2}; \frac{n_1}{2})$.

In general, the Nash equilibria of a DAPP heist game are enumerated by finding the E-set for each M-set, and then finding the equilibria of the game over all E-sets. For convenience, we say that an M-set is a Nash equilibrium if the associated E-set is a Nash equilibrium for the entire game.

7.3 Nash Equilibrium for Multiple Public Offerings

Below we analyze the Nash equilibria of both the single round and double round two-player games. The outcome of the single round game hinges entirely on γ; when $\gamma < \frac{1}{2}$ attackers are incentivized to exploit the vulnerability, otherwise they look to short TOK exclusively. The outcome of the double round game is more complex. Attackers are still incentivized to short TOK exclusively when $\gamma > \frac{1}{2}$, but when $\gamma < \frac{1}{2}$ they are torn between exploiting immediately or shorting in the first round followed by exploiting in the second. We show that their preference can be swayed by adjusting the relative sizes of the first and second round offerings.

7.3.1 Single-round game. In a single round game, each attacker A_j elects to either short $m_{1,j}$ TOK or exploit the vulnerability (but not both). Table 1 shows the payoff for each M-set among two attackers. In general, the optimal strategy for both players depends entirely on the value of γ. We next prove the intuitive result that the best strategy is to short TOK only as long as shorting is more efficient than exploiting ($\gamma > \frac{1}{2}$). More importantly, we show that unique Nash equilibria exist for nearly all values of γ.

THEOREM 5: When $\gamma \neq \frac{1}{2}$, the two-player single round DAPP heist game has a unique Nash equilibrium. Strategy set $(E_{1,1}, E_{1,2})$ (both exploit) is the equilibrium when $\gamma < \frac{1}{2}$, and $(S_{1,1}, S_{1,2})$ (both short) is the equilibrium when $\gamma > \frac{1}{2}$. All strategy sets are equilibria if $\gamma = \frac{1}{2}$.

PROOF: Note that $(S_{1,1}; S_{1,2})$ has extremal payoff $(\frac{n_1}{2}; \frac{n_1}{2})$ because the optimal approach for each attacker is to short as many TOK as possible given that they have both committed to shorting. M-set $(S_{1,1}; E_{1,2})$ has extremal payoff $(\gamma n_1; n_1(1-\gamma))$ because if A_1 plans to short TOK and A_2 plans to exploit, then the optimal strategy for A_1 is to short as many TOK as possible; and for similar reasons, $(E_{1,1}; S_{1,2})$ has extremal payoff $(n_1(1-\gamma); \gamma n_1)$. Finally, strategy set $(E_{1,1}; E_{1,2})$ is not actually meta, therefore its extremal payoff remains equal to $(\frac{n_1}{2}; \frac{n_1}{2})$. There are four types of M-sets; we consider each below:

- *Case 1: Both short TOK.* $(S_{1,1}; S_{1,2})$ is a Nash equilibrium when $\frac{n_1}{2} \geq n_1(1-\gamma)$, which implies $\gamma \geq \frac{1}{2}$; or more intuitively, this equilibrium applies when $\mathcal{D}$ strongly limits the daily ETH withdrawal limit.
- *Case 2: Both exploit.* $(E_{1,1}; E_{1,2})$ can be a Nash equilibrium only if $\frac{n_1}{2} \geq \gamma n_1$, which implies that $\gamma \leq \frac{1}{2}$; or more intuitively, this equilibrium applies when $\mathcal{D}$ does impose only weak limits on daily ETH withdrawal limit.
- *Case 3: A_1 shorts; A_2 exploits.* $(S_{1,1}; E_{1,2})$ is a Nash equilibrium if $\gamma n_1 \geq \frac{n_1}{2}$ and $n_1(1-\gamma) \geq \frac{n_1}{2}$. The two constraints can be satisfied simultaneously only if $\gamma = \frac{1}{2}$.
- *Case 4: A_1 shorts; A_2 exploits.* M-set $(E_{1,1}; S_{1,2})$ is symmetric with $(S_{1,1}; E_{1,2})$. Therefore, it is also a Nash equilibrium if $\gamma = \frac{1}{2}$.

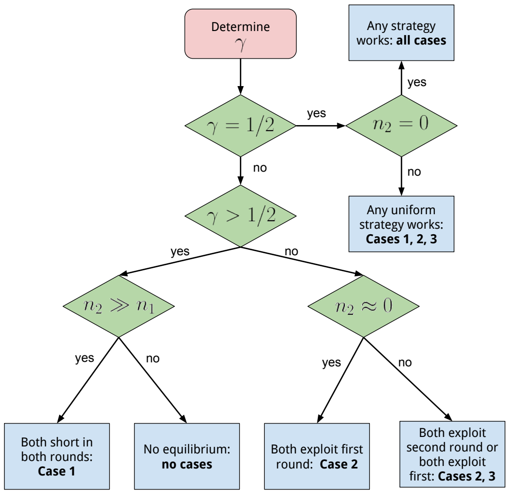
■

Best overall strategy. Theorem 5 shows that the equilibria of the single round game depend entirely on the value γ. Moreover, the game has a unique equilibrium in all circumstances except when $\gamma = \frac{1}{2}$, which is equivalent to equal efficiency for shorting TOK as exploiting the vulnerability. Shorting TOK is preferable to losing all ETH to theft, but in the single round game our only influence on attacker behavior is to attempt to increase γ as much as possible. This is a fundamental limitation of single round games. Next we explore how the attackers' incentives change with the introduction of a second public offering.

7.3.2 Double-round game. In general, the DAPP heist game can be played in n rounds. Here we analyze a double-round game, which is simpler to discuss. Our principal result is that the introduction of a second public offering can induce attackers to either exploit in the first round or short TOK in the first round before exploiting in the second. In either case, fewer ETH will be stolen than if there had been

		$\mathbf{A_2}$		
		$[S_{1,2};S_{2,2}]$	$E_{1,2}$	$[S_{1,2};E_{2,2}]$
	$[S_{1,1};S_{2,1}]$	$(M_{2,1};M_{2,2})$	$(\gamma m_{1,1};n_1-\gamma m_{1,1})$	$(m_{1,1}+\gamma m_{2,1};N_2-m_{1,1}-\gamma m_{2,1})$
$\mathbf{A_1}$|	$E_{1,1}$	$(n_1-\gamma m_{1,2},\gamma m_{1,2})$	$(\frac{n_1}{2};\frac{n_1}{2})$	$(n_1-\gamma m_{1,2};\gamma m_{1,2})$
	$[S_{1,1};E_{2,1}]$	$(N_2-m_{1,2}-\gamma m_{2,2};m_{1,2}+\gamma m_{2,2})$	$(\gamma m_{1,1};n_1-\gamma m_{1,1})$	$(\frac{N_2+m_{1,1}-m_{1,2}}{2};\frac{N_2-m_{1,1}+m_{1,2}}{2})$

Table 2: Payoff matrix for the 2-attacker, double-round public offering DAPP heist game. Because the matrix is symmetric, we gray out the upper diagonal for clarity. Figure 8 is a flow chart for determining the Nash equlibria given γ, n_1, and n_2.



Determine γ
$\gamma = 1/2$
yes
$n_2 = 0$
yes
Any strategy works: all cases
no
Any uniform strategy works: Cases 1, 2, 3
no
$\gamma > 1/2$
yes
no
$n_2 \gg n_1$
$n_2 \approx 0$
yes
no
yes
no
Both short in both rounds: Case 1
No equilibrium: no cases
Both exploit first round: Case 2
Both exploit second round or both exploit first: Cases 2, 3

Figure 8: Flowchart for determining the Nash equilibria of the two-player double round game. When $\gamma = \frac{1}{2}$, most strategy sets are equilibria. If $\gamma > \frac{1}{2}$, then Case 1 (both short both rounds) is an equilibrium provided that n_2 is much larger than n_1. When $\gamma < \frac{1}{2}$ there is a unique equilibrium for Case 2 (both exploit in the first round) provided n_2 is close to zero. Otherwise, Case 3 (short, then exploit) is also an equilibrium for large values of n_2.

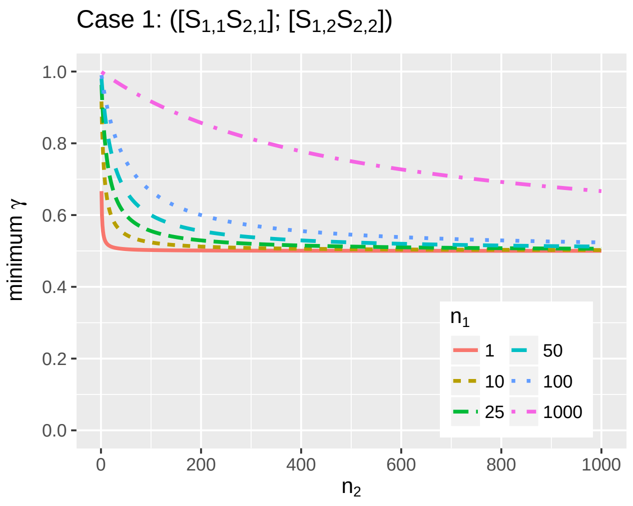


Figure 9: Lower bounds on values of γ that will make Case 1 (both short exclusively) a Nash equilibrium for various choices of n_1 and n_2 in the two-played double round game. Generally, if n_2 is much smaller than n_1, then γ must be very large, but as n_2 becomes much larger than n_1, any choice where $\gamma > \frac{1}{2}$ creates an equilibrium.

a single public offering. And if attackers exploit in the first round, fewer TOK holders will suffer losses from the theft. The following theorem identifies the Nash equilibria of the game. Unfortunately, for the double round game there exist two equilibria for small γ. To address this problem we later conduct formal equilibrium selection.

THEOREM 6: When $\gamma > \frac{1}{2}$ and n_2 sufficiently large, strategy set $([S_{1,1},S_{2,1}];[S_{2,1},S_{2,2}])$ (both short both rounds) is the only Nash equilibrium of the two-player double round DAPP heist game. And a single equilibrium also exists at $(E_{1,1};E_{1,2})$ (both exploit in round 1) for $\gamma < \frac{1}{2}$, unless n_2 is sufficiently larger than 0, in which case $([S_{1,1},E_{2,1}];[S_{2,1},E_{2,2}])$ (both exploit in round 2 after shorting in round 1) is also an equilibrium.

PROOF: There are three types of meta strategies for each attacker in the double-round game: *(i)* the attacker shorts some of the TOK offered in each of the two rounds but never exploits the vulnerability; *(ii)* the attacker shorts some of the TOK offered in round 1, and then exploits in round 2; and *(iii)* the attacker never shorts any TOK and exploits in round 1. Table 2 shows the payoff for each strategy set among two attackers. Because the payoff matrix is symmetric, we search for Nash equilibria on and below the diagonal of the matrix. Appendix C gives the extremal payoff for each of the matrix entries of interest. Those results can be thought of as defining an extremal payoff matrix parameterized only by γ, n_1, and n_2, which defines the maximum payoff that each M-set can achieve for any combination of the $m_{i,j}$.

Having derived the extremal payoff matrix for all M-sets at or below the main diagonal, we now turn our attention to searching for Nash equilibria. There are five major types of M-sets to consider:

- *Case 1: Both short both rounds.* Based on the extremal payoff matrix, $([S_{1,1},S_{2,1}];[S_{1,2},S_{2,2}])$ is a Nash equilibrium when

$\frac{N_2}{2} \geq n_1(1-\gamma)$ and $\frac{N_2}{2} \geq N_2 - \frac{\gamma n_1}{2} - \gamma n_2$. Together these constraints imply that $\gamma \geq \max(\frac{n_1-n_2}{2n_1}, \frac{n_1+n_2}{n_1+2n_2})$.

- *Case 2: Both exploit in round 1.* According to the extremal payoff matrix, M-set $(E_{1,1}; E_{1,2})$ is a Nash equilibrium only if $\frac{n_1}{2} \geq \gamma n_1$, which implies that $\gamma \leq \frac{1}{2}$.
- *Case 3: Both short in round 1, then exploit in round 2.* The extremal payoff matrix indicates that $([S_{1,1}, E_{2,1}]; [S_{1,2}, E_{2,2}])$ is a Nash equilibrium when $\frac{N_2}{2} \geq n_1(1-\gamma)$ and $\frac{N_2}{2} \geq \frac{n_1}{2} + \gamma n_2$. This requires that $\frac{n_1-n_2}{2n_1} \leq \gamma \leq \frac{1}{2}$.
- *Case 4: One exploits in round 1, the other shorts.* According to the extremal payoff matrix, $([E_{1,1}]; [S_{1,2}, E_{2,2}])$ is a Nash equilibrium when $n_1(1-\gamma) \geq \frac{N_2}{2}$; $n_1(1-\gamma) \geq N_2 - \frac{n_1}{2} - \gamma n_2$, and $\gamma n_1 \geq \frac{n_1}{2}$. The second constraint forces $n_1 > n_2$ for all γ, and the other two imply that
$$\frac{1}{2} \leq \gamma \leq \frac{n_1 - n_2}{2n_1},$$
which means that the extremal strategy set can be a Nash equilibrium only if both $\gamma = \frac{1}{2}$ and $n_2 = 0$.
- *Case 5: Both short in round 1, then one exploits in round 2.* The extremal payoff matrix indicates that M-set $([S_{1,1}, E_{1,2}], [S_{1,2}, S_{2,2}])$ is a Nash equilibrium only if $N_2 - \frac{n_1}{2} - \gamma n_2 \geq n_1(1-\gamma)$, $N_2 - \frac{n_1}{2} - \gamma n_2 \geq \frac{N_2}{2}$, $\frac{n_1}{2} + \gamma n_2 \geq n_1(1-\gamma)$, $\frac{n_1}{2} + \gamma n_2 \geq n_1(1-\gamma)$, and $\frac{n_1}{2} + \gamma n_2 \geq \frac{N_2}{2}$. In this case the first constraint forces $n_1 > n_2$ for all γ; and the second and fourth constraints force $\gamma = \frac{1}{2}$. This value for γ along with the first and third constraints show
$$\frac{n_1}{2(n_1+n_2)} \leq \frac{1}{2} \leq \frac{2n_2 - n_1}{2(n_2 - n_1)},$$
which is only valid when $n_2 = 0$.

■

The double round game has a more complicated landscape of equilibria than the single round game. In general, the values for γ, n_1, and n_2 are all determinants of the equilibria. Figure 8 consolidates the results from Theorem 6 into a flowchart depicting the regions where various M-sets achieve a unique equilibrium. In such regions we say that a strategy set *dominates*. The region of dominance for Case 1 (both short in both rounds) is particularly complex. Not surprisingly, it will dominate only when $\gamma > \frac{1}{2}$, which means that it is easier for the attacker to short TOK than to exploit. However, the precise interval for γ where this M-set achieves equilibrium depends in a complex way on the relative sizes of n_1 and n_2. Figure 9 demonstrates how the lower bound on γ changes with the sizes of the public offerings. In general, if n_2 is much larger than n_1, then Case 1 dominates for all $\gamma > \frac{1}{2}$. But if the value of n_2 is significantly smaller than n_1, then Case 1 dominates only in the vicinity of $\gamma = \frac{1}{2}$.

A similar phenomenon occurs when $\gamma < \frac{1}{2}$. Here, Case 2 (both exploit in the first round) dominates for all $\gamma < \frac{1}{2}$ so long as the second public offering is zero. However, once the size of the second public offering increases substantially, Case 3 (both short in round 1 and exploit in round 2) also manifests a Nash equilibrium. This behavior can be explained intuitively as the second round offering becoming so large that attackers who were previously only incentivized to exploit immediately in round 1, must now consider the benefit of shorting in round 1 and waiting until round 2 to exploit.

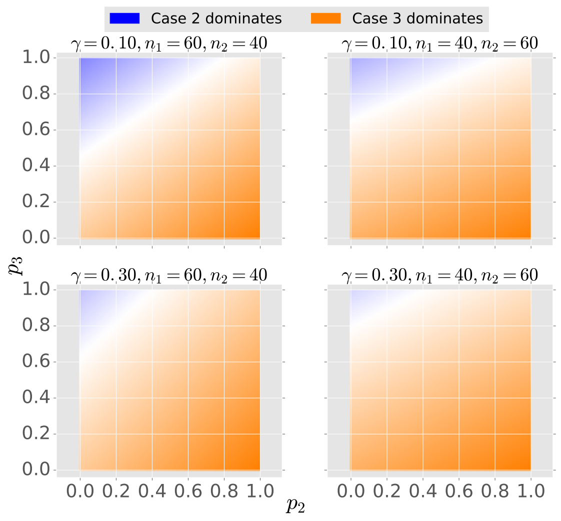


Figure 10: Dominance of equilibria according to the risk function, Equation 12, for various values of γ, n_1, and n_2. Values p_2 and p_3 are the probabilities that an attacker will defect from Case 2 and Case 3 respectively. The color and intensity of each point (p_2, p_3) indicates the sign and magnitude of $R(p_2, p_3)$. White points indicate that $R(p_2, p_3) = 0$, dark blue points suggest $R(p_2, p_3) \gg 0$, where Case 2 dominates, and dark orange points appear when $R(p_2, p_3) \ll 0$, where Case 3 dominates.

Indeed, in terms of potential payoff alone, the extremal payoff for Case 3 is always greater than that for Case 2 as long as $n_2 > 0$. However, waiting to exploit is a riskier strategy; therefore, it is not immediately clear which of these two equilibria should be most attractive to an attacker.

Risk analysis. We have seen that there exist two competing equilibria when $\gamma < \frac{1}{2}$ and $n_2 \neq 0$. The M-set where both attackers short in the first round, then exploit in the second (Case 3) has higher extremal payoff, but is riskier than the other equilibrium where both attackers exploit in the first round (Case 2). Label the former M-set $\mathcal{S}_3$ and the latter $\mathcal{S}_2$. We wish to gain further insight into which meta strategy each attacker is likely to choose. Define the *risk* of $\mathcal{S}_i$ to be the maximum loss in extremal payoff for A_j associated with any other attacker A_k making the *defection* from $\mathcal{S}_i$ to another strategy with least payoff for A_j. Let y_j and y_j' be the payoff for attacker A_j before and after, respectively, attacker A_k defects from $\mathcal{S}_i$.

The *risk function* $R(p_2, p_3)$ between $\mathcal{S}_2$ and $\mathcal{S}_3$ is defined as

$$\begin{aligned} &R(p_2, p_3) \\ &= ((1-p_2)y_2 + p_2 y_2') - ((1-p_3)y_3 + p y_3') \\ &= (1-p_2)\frac{n_1}{2} + p_2\gamma - (1-p_3)\frac{N_2}{2} - p_3\gamma n_1 \qquad (12) \end{aligned}$$

where p_i is the probability of defection from $\mathcal{S}_i$. Attackers will opt for $\mathcal{S}_2$ when $R(p_2, p_3) > 0$ and $\mathcal{S}_3$ when $R(p_2, p_3) < 0$. When $R(p_2, p_3) = 0$ we say that the equilibria are *risk neural*.

Figure 10 shows the direction and magnitude of the risk function (Equation 12) across all possible values p_2 and p_3 and for a selection of parameters γ, n_1, and n_2. Dark blue points indicate that S_2 (Case 2) dominates while dark orange points indicate S_3 (Case 3) dominates. In general, S_3 dominates S_2 for most defection probabilities. Only when p_3, the probability of defecting from S_3, is very high or p_2, the probability of defecting from S_2, is very low, does S_2 begin to dominate. And even then, only relatively small values of γ yield large regions of dominance for S_2. This behavior makes sense because small values of γ indicate extremely poor payoff when shorting TOK, which incentivizes attackers to exploit immediately in round 1. Finally, there is a tendency for S_2 to dominate when $n_1 > n_2$ and for S_3 to dominate otherwise. This also makes sense because attackers will naturally want to exploit right away if the bulk of the value is available in the first round, but are incentivized to wait if substantially more TOK will be made available in round 2.

Best overall strategy set. If shorting TOK is more efficient than exploiting the vulnerability, $\gamma > \frac{1}{2}$, and n_2 is sufficiently large (see Figure 9), then there exists a unique Nash equilibrium corresponding to Case 1: both attackers short both rounds. In terms of protecting the ETH in $\mathcal{D}$ from theft, Case 1 is the ideal behavior because, under the extremal strategy, all ETH is legitimately withdrawn from the DAPP before the vulnerability is exposed. Thus it makes sense for the DAPP designer to attempt to raise γ as much as possible in an attempt to bring it above $\frac{1}{2}$. There are two major actions the designer can take to help increase γ: *(i)* limit daily aggregate withdrawals, which might slow the rate of theft; and *(ii)* ensure that there is a vibrant futures market in order to make shorting easier for the attacker. Assuming that it is possible to raise γ above $\frac{1}{2}$, Case 1 becomes a stronger equilibrium to the extent that n_2 exceeds n_1. So the DAPP designer should also seek to make the size of the second public offering significantly larger than the size of the first.

No matter what counter measures are taken, there might still be vulnerabilities that can be exploited significantly faster than TOK can be (profitably) shorted. In these cases $\gamma < \frac{1}{2}$ and there exist two Nash equilibria: Case 2 and Case 3, with the latter being an equilibrium only when $n_2 \geq n_1(1 - 2\gamma)$. Thus for $\gamma < \frac{1}{2}$ we must accept that the vulnerability will eventually be exploited, but we can still mitigate the size of the exploitation by understanding attacker defection probabilities and tuning the balance between n_1 and n_2. Note that, in terms of the amount of ETH stolen from $\mathcal{D}$, if $n_1 < n_2$, then Case 2 is preferable because less than $\frac{n}{2}$ will be stolen, but if $n_1 > n_2$ then Case 3 is preferred because under the extremal strategy, all the TOK offered in round 1 will be shorted and the associated ETH in $\mathcal{D}$ will be legitimately withdrawn.

It is reasonable to estimate low values for p_2 and high for p_3 because Case 2 is inherently lower risk than Case 3. Figure 10, or a similar instrument, can be used to determine an appropriate balance between n_1 and n_2 given estimates for defection probabilities. For example, if $p_2 < 0.1$ and $p_3 < 0.65$, then it is possible to make n_1 somewhat larger than n_2 and still incentivize Case 3, even for large γ. However, if $p_3 > 0.9$, then it is wise to make n_2 larger than n_1 because Case 2 will dominate for almost any balance of public offerings and, hence it is preferable to make n_1 a small as possible in order to minimize the amount of ETH that can be exploited. Similar reasoning holds for mitigating theft under Case 3. If we have reason to believe that $p_2 < 0.4$ and $p_3 < 0.6$, then it makes sense to ensure that n_2 is less than n_1, because Case 3 will dominate for nearly every value of γ.

Analysis of risk is a good technique to use when it is possible to make some guess about the probabilities of defection. However, this is not always possible. Under such circumstances, a reasonable alternative approach is to opt for an even split, $n_1 = n_2$. By splitting the TOK offered between rounds evenly, we can ensure that no more than half of the ETH deposited will be stolen from $\mathcal{D}$, no matter what equilibrium the game falls into.

8 FUTURE WORK

In future work we plan to extend and improve upon both the DAPP insurance and gated IPO mechanisms. There are several interesting avenues to explore in the development of DAPP insurance. First, we seek a more robust technique for determining when to close λ-SPs. While using fixed price threshold α can guarantee the minimum ETH recovered, it is far from ideal: there will always exist some size theft that will fail to push λ-SP value across the threshold, recovering no ETH; and the mechanism does not explicitly account for the passage of time, making it incapable of accounting for the probability of margin call. Second, it is unclear how the recovery processor will behave in the presence of market manipulation. For example, there exist techniques whereby manipulators can profitably trigger cascading margin calls. How should the DAPP adjust its required leverage in order to avoid having its λ-SPs liquidated?

Gated IPOs can also be improved in several ways. First, we have only analyzed single and double round IPOs, but any finite number of issuance rounds will incentivize vulnerability exploitation in the final round. Moreover, because ETH can be redeemed for TOK at any time, the TOK supply will be deflationary over time. Thus it would be interesting to explore secure ways of extending the DAPP Heist Game to infinite rounds. Second, the DAPP insurance and gated IPO mechanisms can, in principle, be deployed together. However, the presence of the insurance mechanism will change the payoffs in the DAPP Heist Game. Specifically, because of the insurance, the trade price of $\frac{\text{ETH}}{\text{TOK}}$ will never drop to zero, which means that TOK shorted during an IPO will not realize as much value when the vulnerability is exploited. Thus we are interested in determining how the dynamics of both mechanisms change when deployed simultaneously.

9 CONCLUSION

We have presented two generic mechanism that can be used to insure the ether holdings of DAPPs that are similar to The DAO. Our mechanisms are embedded into and controlled by the software itself. Our first approach is to charge a small withdrawal fee, which is used to purchase futures contracts that hedge against a drop in the price of tokens issued by the DAPP. We show that, even in the event that all ether holdings are stolen from the DAPP, the majority of the ether can typically be restored with high probability. More specifically, we find that in order to ensure that a large fraction of assets are recoverable it is necessary that either the volatility of futures contracts remains relatively low or the withdrawal fee remains high. Our second approach is based on multi-round, gated public offerings. We show that it is always possible in a double-round offering to either encourage attackers to exploit the vulnerability early, which reduces the total ETH stolen, or encourage attackers to delay exploitation and short TOK in the market instead, which also reduces the total ETH stolen from the DAPP as instead the loss is transferred to the market.

10 ACKNOWLEDGEMENTS

We would like to thank participants in the NSPW workshop for their peer review, lively discussion, and much helpful feedback. We also wish to thank David Thibodeau for his careful read of our manuscript, insightful feedback, and identification of technical errors.

REFERENCES

[1] BitMex. https://www.bitmex.com. (November 2016).
[2] Etheria. http://etheria.world. (November 2016).
[3] Gnosis. https://www.gnosis.pm. (November 2016).
[4] OkCoin. https://www.okcoin.com/. (November 2016).
[5] Oraclize.it. http://www.oraclize.it. (November 2016).
[6] SafeMarket. https://safemarket.github.io. (November 2016).
[7] Bitcoin Smart Contracts. https://en.bitcoin.it/wiki/Contract. (January 2017).
[8] Digix Dai. https://github.com/makerdao/docs/blob/master/Dai.md. (July 2017).
[9] Digix Dao. https://www.dgx.io/dgd. (July 2017).
[10] Digix Global. https://www.dgx.io. (July 2017).
[11] Tether. https://tether.to. (July 2017).
[12] Yunbi. https://yunbi.com/markets/dgdcny?lang=en-US. (July 2017).
[13] Ross Anderson and Tyler Moore. 2006. The Economics of Information Security. *Science* 314, 5799 (2006), 610–613. https://doi.org/10.1126/science.1130992 arXiv:http://science.sciencemag.org/content/314/5799/610.full.pdf
[14] A. W. Appel. 2016. Modular Verification for Computer Security. In *Proc. IEEE Computer Security Foundations Symposium (CSF)*. 1–8. https://doi.org/10.1109/CSF.2016.8
[15] Rainer Böhme and Galina Schwartz. 2010. Modeling Cyber-Insurance: Towards a Unifying Framework.. In *Proc. Workshop on the Economics of Information Security (WEIS)*.
[16] William Bush, Jonathan Pincus, and David Sielaff. 2000. A Static Analyzer for Finding Dynamic Programming Errors. *Software-Practice and Experience* 30, 7 (2000), 775–802.
[17] Vitalik Buterin. Thinking About Smart Contract Security. https://blog.ethereum.org/2016/06/19/thinking-smart-contract-security. (January 2017).
[18] L. Jean Camp and Catherine Wolfram. 2000. Pricing Security. In *Proc. of the CERT Information Survivability Workshop*. 31–39.
[19] Coq development team. Reference Manual version 8.6. https://coq.inria.fr/refman/. (Dec 14 2016).
[20] Michael del Castillo. Ethereum Executes Blockchain Hard Fork to Return DAO Funds. http://www.coindesk.com/ethereum-executes-blockchain-hard-fork-return-dao-investor-funds. (July 2016).
[21] Michael del Castillo. The DAO Attacked: Code Issue Leads to $60 Million Ether Theft. http://www.coindesk.com/dao-attacked-code-issue-leads-60-million-ether-theft. (June 2016).
[22] J. Douceur. 2002. The Sybil Attack. In *Proc. International Workshop on Peer-to-Peer Systems (IPTPS)*. 251–260.
[23] Serge Egelman, Cormac Herley, and Paul C. Van Oorschot. 2013. Markets for Zero-Day Exploits: Ethics and Implications. In *Proc. New Security Paradigms Workshop*. 41–46.
[24] Ethereum. Ethereum Homestead Documentation. http://ethdocs.org/en/latest/. (2017).
[25] Michael Fischer, Nancy Lynch, and Michael Paterson. 1985. Impossibility of distributed consensus with one faulty process. *J. ACM* 32, 2 (1985), 374–382.
[26] Lloyd D. Fosdick and Leon J. Osterweil. 1976. Data Flow Analysis in Software Reliability. *ACM Comput. Surv.* 8, 3 (Sept. 1976), 305–330.
[27] Jens Grossklags, Nicolas Christin, and John Chuang. 2008. Secure or Insure? A Game-theoretic Analysis of Information Security Games. In *Proc. International Conference on World Wide Web*. 209–218.
[28] Benjamin Johnson, Rainer Böhme, and Jens Grossklags. 2011. Security Games with Market Insurance. In *Proc. International Conference on Decision and Game Theory for Security*. 117–130.
[29] Aron Laszka and Jens Grossklags. 2015. Should Cyber-Insurance Providers Invest in Software Security? In *Proc. European Symposium on Research in Computer Security*. 483–502.
[30] Mohammad Hossein Manshaei, Quanyan Zhu, Tansu Alpcan, Tamer Bacşar, and Jean-Pierre Hubaux. 2013. Game Theory Meets Network Security and Privacy. *ACM Comput. Surv.* 45, 3 (July 2013), 25:1–25:39.
[31] Angelica Marotta, Fabio Martinelli, Stefano Nanni, Albina Orlando, and Artsiom Yautsiukhin. 2017. Cyber-insurance Survey. *Computer Science Review* 24 (2017), 35–61.
[32] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf. (May 2009).
[33] Micah Schwalb. 2007. Exploit Derivatives & National Security. *Yale Journal of Law & Technology* 9, 1 (2007).
[34] David Siegel. Understanding The DAO Attack. http://www.coindesk.com/understanding-dao-hack-journalists. (June 2016).
[35] Kyle Torpey. Millions of Dollars Worth of ETC May Soon Be Dumped on the Market. https://bitcoinmagazine.com/articles/millions-of-dollars-worth-of-etc-may-soon-be-dumped-on-the-market-1472567361. (August 2016).
[36] Marko Vukolić. 2015. The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication. In *Proc. Open Problems in Network Security: IFIP WG 11.4 International Workshop*. 112–125.

A BUYING ON MARGIN

Example 1. Suppose a trader brings 10 ETH to a market where a DAPP token sells for $1\frac{\text{ETH}}{\text{TOK}}$. Without the use of leverage, the trader can purchase $\delta = 10$ short position futures contracts. If at the contract's close, the spot rate falls to $0.8\frac{\text{ETH}}{\text{TOK}}$, her profit would be $0.2\text{ETH} \cdot 10 = 2\text{ETH}$, increasing her equity to 12ETH. If instead the spot rate increases to $1.25\frac{\text{ETH}}{\text{TOK}}$ at the contract's close, then she would need to pay out $0.25\text{ETH} \cdot 10 = 2.5\text{ETH}$ to the opposing party, reducing her equity to 7.5 ETH. If the rate increases to $2\frac{\text{ETH}}{\text{TOK}}$, she will owe 10 ETH to the opposing party; therefore, the exchange will enforce a margin call, closing the contracts before the rate increases any further.

Example 2. Now consider the outcomes if the market allows the trader to buy short position futures with leverage of $\lambda = 4$. The trader purchases 40 contracts at a spot price $1\frac{\text{ETH}}{\text{TOK}}$, but covering only 1/4 the equity of each contract. If at the contract's close, the spot rate falls to $0.8\frac{\text{ETH}}{\text{TOK}}$, her profit would be $0.2\text{ETH} \cdot 40 = 8\text{ETH}$, increasing her equity to 18ETH. If instead the spot rate increases to just $1.25\frac{\text{ETH}}{\text{TOK}}$, then she would owe $0.25\text{ETH} \cdot 40 = 10\text{ETH}$ to the opposing party; accordingly, the exchange would enforce a margin call and close the contracts.

B SHORTING IN THE FUTURES MARKET

In addition to shorting TOK in the spot market, the attacker can also open short positions, SPs, if there exists a viable futures market (see Section 5). At first it seems possible that the presence of short futures contracts would offer the attacker an additional avenue for profiting from the exploitation of a vulnerability. Indeed this opportunity is potentially more expedient for the attacker, which is an important advantage. But the following property is also important: if the spot and futures markets are efficient, exhibit supply-demand symmetry, and the depth of the futures market is less than that of the spot market, then purchasing an SP has the same impact on the ETH holdings of $\mathcal{D}$ as does shorting a TOK in the spot market.

Consider the cross-market dynamics associated with opening an SP at a time when both the spot and futures markets are at equilibrium. This means that both the current spot and futures price of $\frac{\text{ETH}}{\text{TOK}}$ are equal to one. The attacker must *sell* a futures contract in order to open an SP. Presumably the sale will cause a small decrease in the trade price of $\frac{\text{ETH}}{\text{TOK}}$ in the futures market. In response to this discount, if an arbitrager expects that she can earn a profit, she will buy a futures contract to open a corresponding LP (long position). Because of the symmetry of supply and demand, the trade price on the futures market will return to equilibrium. Now in order for the arbitrager to hedge her position, she will short a TOK on the spot market, which we have argued in Section 7.1 will result in a TOK being traded for ETH in $\mathcal{D}$, and will result in a return to equilibrium in the spot market.

In the scenario above, the arbitrager will earn a profit only if the difference between the price she paid to open the LP (at discount) and the price she received to sell TOK (also at a discount) is positive. Because the depth of the spot market is assumed to be greater than the depth of the futures market, there are more willing buyers and sellers of TOK in the spot market than there are buyers and sellers of futures contracts in the futures market. The effect of this discrepancy in depth is that the purchase or sale of a futures contract will tend to move the futures trade price more than the purchase or sale of a TOK moves the index price. Thus, the arbitrager can expect to profit in the scenario above. If we were to relax the market depth assumption, an arbitrager would eventually step in, but she would wait for the attacker to open multiple SPs before she opened a single opposing LP. In effect, the attacker would be able to open multiple SPs and only one TOK would ultimately be drained from $\mathcal{D}$. The analysis we perform in Section 7 could be modified to account for this by simply increasing the payoff for shorting TOK.

Note that depending on the depth of market (both futures and spot), an attacker must short slowly in order to allow the market to return to equilibrium between shorts. Otherwise, if the attacker attempts to short very quickly, then either the TOK shorted or SPs sold will be significantly discounted, cutting into his potential profit.

C EXTREMAL PAYOFF FOR THE DOUBLE ROUND GAME

Below we detail the extremal payoff for each of the matrix entries of interest in the double round DAPP heist game.

- $([S_{1,1}, S_{2,1}]; [S_{2,1}, S_{2,2}])$: If each attacker always plans to short, then the optimal strategy for both is to short as much TOK as possible in every round. Hence the extremal payoff is $(\frac{N_2}{2}; \frac{N_2}{2})$.
- $(E_{1,1}; E_{1,2})$: The strategy set where both attackers exploit in the first round is not meta. Therefore, the extremal payoff remains $(\frac{n_1}{2}; \frac{n_1}{2})$.
- $([S_{1,1}, E_{2,1}]; [S_{1,2}, E_{2,2}])$: For A_1, the payoff vector is an increasing function of $m_{1,1}$ and a decreasing function of $m_{1,2}$. The opposite is true for A_2. Thus, the extremal strategy involves attacker j attempting to maximize $m_{1,j}$ at the expense of $m_{1,3-j}$, i.e. $m_{1,1} = m_{1,2} = \frac{n_1}{2}$. It follows that the extremal payoff is $(\frac{N_2}{2}, \frac{N_2}{2})$.
- $(E_{1,1}; [S_{1,2}, S_{2,2}])$: When A_2 elects to short in round 1 but A_1 exploits, it is in the best interest of A_2 to short as many TOK as possible. Therefore the extremal payoff is $(n_1(1-\gamma), \gamma n_1)$.
- $([S_{1,1}, E_{2,1}]; E_{1,2})$: The extremal strategies for each attacker are analogous (but opposite) to $(E_{1,1}; [S_{1,2}, S_{2,2}])$. Hence the extremal payoff is also $(n_1(1-\gamma); \gamma n_1)$.
- $([S_{1,1}, E_{2,1}]; [S_{1,2}, S_{2,2}])$: The payoff for A_1 in this meta strategy is a decreasing function of both $m_{1,2}$ and $m_{2,2}$; and it is an increasing function of those variables for A_2. Because A_1 will not exploit and short in the same round, it is clear that A_2 will fully maximize $m_{2,2}$, i.e. $m_{2,2} = \gamma n_2$. On the other hand, both attackers short in round 1. Thus under the extremal strategy, $m_{1,1} = m_{1,2} = \frac{n_1}{2}$. This implies that the extremal payoff is $(N_2 - \frac{n_1}{2} - \gamma n_2; \frac{n_1}{2} + \gamma n_2)$.