Platonic.Systems

for Project Ardana

Audit

Authors

Quinn Dougherty `quinn.dougherty@platonic.systems`

Foo Bar `foo.bar@platonic.systems`

Supervisor

Isaac Shapira `isaac.shapira@platonic.systems`

October 6, 2021- 23:37

Table of Contents

I. Preamble

The audit is a preliminary effort to compensate for the fact that proper formal verification before launch is infeasible.

An audit means many things. Let's be precise about what we mean by audit in this document.

Definition I.0.1 (Audit)**.** *An **audit** is a document provided to the community to guide them in taking informed risk.*

Definition I.0.2 (Community)**.** *A **community** consists of liquidity providers, investors, swappers, arbitrageurs, governance token holders, and neighboring members/projects of the ecosystem.*

I.0.1. Desiderata

- **The community is the audience. The community is the customer.**
- I think we want to make an explicit demarcation of risks for whom. Some risks only effect Ardana internally, other risks effect the collective of Dana holders, other risks effect our neighbors in the ecosystem. I think there's a difference between risks we want to eliminate and risks we want to empower the community to take. I'm sort of imagining profiling these as separate *axes* of risk. The idea here is that I'm frustrated with the literature because it almost feels like it uses the word "attack" anytime someone loses money. This is not adequate- sometimes you just gambled and lost, sometimes the "attacker" followed the rules of the mechanism as designed. There's a morality/values question to each attack, and **the audit needs to provide disambiguation and clarity without taking sides on each values/morality question**.
- The audit is **as much as was possible to do before launch time, not exhaustive**.

I.1. Considerations

In this chapter we look at broad concepts and decisions and provide context into the way the team is thinking about them. This section should add indirect value to the process of taking informed risks.

I.2. Attacks

In this chapter we profile threat models, attack vectors, vulnerabilities; mostly on the economic and mechanism design levels, but occasionally on the software implementation level.

This audit will take on a bit of a code-is-law opinion; many things which are called "attacks" are in fact people using mechanisms as designed. However, it is still the responsibilty of a platform (such as a DeX) to help the community make informed decisions about risk, even when the risk concerns unforeseen behaviors of a protocol or implementation.

Philosophically, be wary of morally charged language in the overall literature. It often implies that an attack is carried out by a summary enemy of the entire ecosystem, that the ecosystem is victimized, when clearer thinking shows that a small team or platform was the sole victim.

II. Considerations

II.1. On datastream integrity

There are a couple places in the Ardana ecosystem that suggest **interaction with live datastreams**. A variety of software concerns such as API integration practices or keeping a stream open are ignored in the current document.

One of these such places is the oracle/bot from the governance layer. The oracle is to consume *third party* price data from something like coinbase, binance, etc. and produce transaction signatures.

The second such place is a proposal Morgan and I discussed on discord[1] for DEX subpool size limits. If we choose to do something functional/dynamic, rather than constant/static, for the subpool size limits the most principled choice is to *infer* them from data. > We can determine it empirically using data such as the data cited here[2] and analytical formulas for determining the pool sizes required to reach slippage targets for different swap sizes (Morgan)

While it's absolutely an option to download a static dataset once (and refresh it every few months or so), the natural question is *do we better serve the project by implementing online learning?* Here I am assuming that somewhere in the literature a formula is written down, making the actual model dead simple. But since the possibility of live datastream has been floated, I want to enumerate some of the pitfalls.

I will provide a few threatmodels that arise when a datastream is interacted with "online."

II.1.1. Threatmodel 1: third parties compromised

The idea is *we do not trust the third party data sources*. Attackers here fall into two camps: those who are trying to corrupt the beliefs of the whole market and those who are trying to corrupt our beliefs in particular.

[1] https://discord.com/channels/844383474676662292/844387861251751978/885557683745349632

[2] https://www.mechanism.capital/liquidity-targeting/

I'm envisioning something like the creation of artificial (even fraudulent) arbitrage opportunities by directly perturbing coinbase/binance's beliefs through hacking, and Ardana's behavior is *downstream* of coinbase/binance's beliefs because of where in our system we interact with their APIs.

Suppose we implement the online learning version of subpool size limit selection. An attacker may be able to arbitrage on pool sizes somehow iff they can force an irrational choice of pool sizes (and if our pool sizes are downstream of coinbase/binance data, all they'd need to do is hack into coinbase/binance).

II.1.1.1. Mitigation

Does coinbase/binance have a notification system that goes out to API users when they detect a breach? If so, we should handle it almost as an error and fallback to the last uncorrupted snapshot when such a notification is retrieved.

Test for agreement between multiple sources. The probability that an attacker compromises multiple data sources in exactly the same way is much lower than the probability that an attacker compromises one of them.

II.1.2. Threatmodel 2: edge case behavior of model

Even if model is dead simple, it could still go off the rails if it got bizarre, unforeseen inputs.

II.1.2.1. Analogy

In a more intricate model, **out-of-distribution robustness[3]** describes the resilience of that model to *shifts in input distribution* or, in the extreme case, an attacker eliciting behavior that the model creator does not want by finding inputs on which the model behaves pathologically. As you can imagine, it is easier for attackers to simply make the model fail (make wrong predictions, for instance) than it is for attackers to target behaviors that they desire (make the model benefit them in some way, for instance).

II.1.2.2. Mitigation

Be extremely liberal in property tests, this is not a time to save testing resources from implausible cases, because implausible cases could be what hurts us.

[3] https://en.wikipedia.org/wiki/Adversarial_machine_learning

Be extremely conservative in input validation/constraints, though beware a lot of inference/data decisions are being made when such constraints are imposed.

II.1.3. Threatmodel 3: positive feedback loop

Optimistically trades and prices made on our DEX system will be a part of the data from coinbase/binance. What does it mean when our behavior shows up in the data from which we derive our behavior?

A potential pitfall here isn't entirely unlike threatmodel 2. Under positive feedback, data can simply be sent off the rails into "edge case behavior" that we didn't think would show up in the operating of our model.

II.1.3.1. Analogy

Fraud detection at Stripe[4] is trained on the prior year's fraud data. The problem is that data is labeled by the earlier iteration of the model, so a perturbation in the model's behavior might lead to a (nonlinearly drastic) perturbation in the new data labels.

II.1.3.2. Mitigation

Via the Stripe example, we can do something called *counterfactual evaluation* which is an algorithm for generating data "as if" our behavior wasn't influencing the data. If this seems important/promising I can workshop with Bassam what this would look like for us.

II.2. Root-finding

Recall the **invariant equation** from the StableSwap Whitepaper (Egorov 2019, 5). In the formalism provided by our Danaswap Whitepaper (Thomas 2021, 3), there exists a function $I : S \to \mathbb{R}$ for contract states S such that $I(s) = 0$ is equivalent to the invariant equation. Danaswap borrows everything from StableSwap to vary between constant-product and constant-sum market-making according to a *leverage* parameter, for which we also accept the suggestion found in (Egorov 2019). Sometimes, we need to hold all balances constant to solve for D (which we call *the invariant*, having the semantics of total amount of coins **when** all coins have equal price). Other times, we

[4] https://youtu.be/rHSpab1Wi9k

consider a k and solve for $B(s)_k$ holding everything else (including D) constant, when $B : S \to \mathbb{R}^n$ is a function assigning in every state a balance to each of n assets (we will think of an $i \in 1..n$ as an *asset label*).

We define the **invariant polynomial** $n+1$ times like so

Definition II.2.1 (Invariant polynomials)**.**

$$I_D := D \mapsto D^{n+1} + (A + n^{-n})n^{2n}(\Pi B(s)_i)D + -An^{2n}(\Pi B(s)_i)\Sigma B(s)_i$$
$$\forall k \in 1..n, I_k := B(s)_k \mapsto B(s)_k^2 + \left(\Sigma_{i\neq k}B(s)_i + (\frac{1}{An^n} - 1)D\right)x_k + \frac{-D^{n+1}}{An^{2n}\Pi_{i\neq k}B(s)_i}$$

The derivations beginning with (Egorov 2019) are in Appendix A.

We think the invariant equation is best represented as polynomials set to zero, depending on what you're solving for, for the following reasons

1. **Characterize the roots in terms of existence and uniqueness**. It can be shown that there is exactly one nonnegative real root for I_D and each I_k, and we'd like the onchain code to be close to the form that makes this easy to see.
2. **Trivially reason about derivatives**. Without my algebraic choices the derivatives (for Newton's method) are harder to see.
3. **Shrink the arithmetic tree size**. Leaving χ in a blackbox has the advantage of the codebase being able to plug in different leverage coefficients in the future just by supplying the leverage coefficient and its derivative. However, this puts more on the stack than is necessary. I haven't done any formal benchmarking of this, but I currently believe the invariant polynomials in these forms are simpler trees and should therefore result in lower fees.
4. **Increase our ability to reason about alternatives to Newton's method**. For example, looking at this problem from a companion matrix point of view becomes possible when we have formal polynomials.

II.2.1. Newton's algorithm

In Curve's implementation of StableSwap, they use Newton's algorithm for root finding (Wikimedia 2021), so that's the first iteration of our codebase.

When the derivative can be found in a neighborhood of zero, Newton's method does not enjoy convergence guarantees.

II.3. Scalar types

II.3.1. Onchain components

We use PlutusTx.Rational[5] to represent numbers in the Danaswap contract. As of this writing, tolerance (number of decimals needed to evaluate equality) is set to `30`.

For the smart contract, we require calculations which are highly precise and can handle very large numbers and can be reproduced exactly across different hardware. Using FLOPs (floating point operations) is not compatible with these requirements. We are not able to determine exactly how big or how precise our numbers need to be, so we cannot say that FLOPs allow for enough size and precision. We can say, however, that FLOPs are implemented slightly differently on different hardware and results may not be reproducible across different hardware. Additionally, FLOPs are not allowed to be used in Plutus on-chain code. These are the constraints which do not allow us to use `Double` to represent numbers in the smart contract.

II.3.2. Offchain components

[5] https://github.com/input-output-hk/plutus/blob/master/plutus-tx/src/PlutusTx/Ratio.hs

III. Attacks

III.1. Vampire attack

III.1.1. Definition

Definition III.1.1 (Vampire attack)**.** *Let Π and Π' be similar protocols, but Π launched and attracted investors and customers earlier, and Π' is somehow derivative of Π. Suppose Π' competes with Π such that Π' makes parameter choices or other measures to become more attractive to investors or customers than Π. A* ***vampire attack*** *is defined as the migration of value (liquidity or other assets) out of Π into Π'.*

III.1.2. The literature

Consult a selection of stories about vampire attacks.

- $\Pi' =$ `SushiSwap`; $\Pi =$ `UniSwap`[1]. SushiSwap was in fact a fork of UniSwap's code, and they provided incentives that directly targeted UniSwap investors and liquidity providers. This is the canonical notion of a vampire attack, with what appears to be the most written about it because of it's scale of impact and how early on the DeFi scene it was found. Our present definition is generalized for analysis that applies outside of the specific conditions here.

- $\Pi' =$ `Swerve`; $\Pi =$ `Curve`[2]. The term "vampire" does not occur in this article, but blaize.tech[3] considers it to be a vampire attack. By forking Curve, Swerve offered a platform very similar to Curve's, and became competitive in TVL in a matter of days while people pulled out of Curve. There doesn't appear to be anything unique about Curve and Swerve being stablecoin DeXes.

- $\Pi' =$ `Artion`; $\Pi =$ `Opensea`[4]. At current writing it's too early to tell, but it's possible that Artion by providing a platform competitive with Opensea will be

[1] https://youtu.be/UFjXwrCGuog
[2] https://finance.yahoo.com/news/swerve-finance-total-value-locked-075020390.html
[3] https://blaize.tech/services/how-to-prevent-liquidity-vampire-attacks-in-defi/
[4] https://www.coindesk.com/tech/2021/09/24/andre-cronjes-new-nft-marketplace-is-a-vampire-attack-su

considered to have vampire attacked it. Unfolding events for this to be the case would have to be that Artion is successful at the expense of Opensea. My choice to be influenced by a CoinDesk writer's choice to call this a vampire attack is up for debate, but my intention is to be consistent with the ecosystem and the literature and I don't see grounds to exclude this writer from either.

- Extended notes on forks[5].

III.1.2.1. Major takeaways

- Lack of vampire attack stories in the Cardano ecosystem is, according to my analysis, not a by-construction property of Cardano. I.e. it's a matter of time.

- Forking a codebase is often used as evidence in favor of the accusation that a given Π' conducted a vampire attack, though forking is not an intrinsic property of the attack.

III.1.3. Scenario: reputational damage if we're considered Π'

Are there competing DeXs that beat us to market that could accuse us of vampire attacking them? Imagine if a bunch of Curve investors pull out their liquidity, exchange it for `ADA` on Coinbase, and start playing Danaswap. Would Curve think of that like a vampire attack?

III.1.4. Scenario: value siphoned out if we become Π

Suppose another DeX for stablecoins launches with an incentive structure more attractive to our commmunity than our own.

Suppose further that, having open sourced the DeX contract code, this competitor copies our onchain code, and makes offchain code similar to ours themselves.

III.1.4.1. Mitigations

- Keeping the DeX code closed source. This is a minor payout in risk reduction

[5] https://newsletter.banklesshq.com/p/fork-defense-strategies-in-defi

- Peg fee parameters to democracy via governance token holders (so we don't have to worry about the wants and needs of the community not being met, we don't have to worry about competitors siphoning them away from us).

III.1.5. Conclusion

In any kind of market, participants take on the unavoidable risk of competitors showing up with better rates. The factor of code forking presented by the open source software context doesn't change this much, and the factor of parameters like fees presented by the economics of DeFi context doesn't either.

III.2. Flashloans

Flashloans are associated with something like `$136M`[6] in[7] losses[8].

Ethereum offers flash loans because they have **multi-step atomic transactions**. Cardano does not have these. So in spite of `Aada`[9]'s claims, we are not expecting flash loans to enter the Cardano ecosystem at this time. There may be lessons from the attacks in the reports, but they are not entirely straightforward. The question is: is there anything *unique* introduced by the flash loan mechanism? The auditing research opportunity here is not a huge priority.

III.2.0.1. Action: monitoring Cardano for developments in multistep atomic transactions

Project Ardana will be monitoring the evolution of Cardano, because we believe that if multistep atomic transactions are introduced flashloans will be shortly around the corner.

III.2.0.2. In this event, the following mitigation strategy sketches will become urgent

- Onchain code only allow interop from one platform and users, not arbitrary platform.

[6] https://peckshield.medium.com/bzx-hack-full-disclosure-with-detailed-profit-analysis-e6b1fa9b18fc
[7] https://news.bitcoin.com/defi-protocol-harvest-finance-hacked-for-24-million-attacker-returns-2-5
[8] https://www.coindesk.com/markets/2021/05/20/flash-loan-attack-causes-defi-token-bunny-to-crash-ov
[9] aada.finance

- Lending products ought to require to collateralize in one whole transaction ahead of time before.

- Block price manipulation by disallowing mid-transaction information from updating prices.

IV. Appendices

IV.1. Appendix A: invariant polynomial

(Egorov 2019) gives us a way of easing between constant-sum and constant-product market-making by a coefficient χ called *leverage*, which turns out to be a function of D and $B(s)$, where $B : S \to \mathbb{R}^n$[1] is a function assigning in every state a balance to each of n assets. In what follows, let $x = B(s)$ such that $x_j = B(s)_j$ for each asset label j.

$$\chi D^{n-1}\Sigma x_i + \Pi x_i = \chi D^n + \left(\frac{D}{n}\right)^n$$

When χ is a blackbox, there is very little analysis available regarding the existence and behavior of roots or the convergence of any root-finding algorithm. We will forego any gains of abstracting over leverage coefficients, and let $\chi = \frac{A(\Pi x_i)n^n}{D^n}$ before proceeding.

IV.1.1. Derivation of invariant polynomials

First we derive I_D, the polynomial in unknown D.

[1] Balances are strictly positive, so it's not really $\mathbb{R}^n$, however we enjoy some vector space properties in [@DanaswapWhitepaper, p. 6] so we do not constrain the set.

Derivation IV.1.1.

$$\chi D^{n-1}\Sigma x_i + \Pi x_i = \chi D^n + \left(\frac{D}{n}\right)^n$$

$$\frac{A(\Pi x_i)n^n}{D^n} D^{n-1}\Sigma x_i + \Pi x_i = \frac{A(\Pi x_i)n^n}{D^n} D^n + \left(\frac{D}{n}\right)^n$$

$$\frac{A(\Pi x_i)n^n}{D}\Sigma x_i + \Pi x_i = A(\Pi x_i)n^n + \frac{1}{n^n}D^n$$

$$\times D \quad \times D$$

$$An^n(\Pi x_i)\Sigma x_i + (\Pi x_i)D = An^n(\Pi x_i)D + \frac{1}{n^n}D^{n+1}$$

$$-An^n(\Pi x_i)D - \frac{1}{n^n}D^{n+1} \quad - An^n(\Pi x_i)D - \frac{1}{n^n}D^{n+1}$$

$$An^n(\Pi x_i)\Sigma x_i + (\Pi x_i)D - An^n(\Pi x_i)D - \frac{1}{n^n}D^{n+1} = 0$$

$$\frac{-1}{n^n}D^{n+1} + (1 - An^n)(\Pi x_i)D + An^n(\Pi x_i)\Sigma x_i = 0$$

$$\times - n^n \quad \times -n^n$$

$$D^{n+1} - n^n(1 - An^n)(\Pi x_i)D - An^{2n}(\Pi x_i)\Sigma x_i = 0$$

$$D^{n+1} + (A - n^{-n})n^{2n}(\Pi x_i)D + -An^{2n}(\Pi x_i)\Sigma x_i = 0$$

We now have a polynomial in $x \mapsto x^{n+1} + ax + b$ form, for constants a and b which are functions of a balance sheet and the *amplification coefficient* A.

Derivation IV.1.2. $\forall k \in 1..n$,

$$\frac{A(\Pi x_i)n^n}{D^n} D^{n-1} \Sigma x_i + \Pi x_i = \frac{A(\Pi x_i)n^n}{D^n} D^n + \left(\frac{D}{n}\right)^n$$

$$\frac{An^n}{D}(\Pi_{i \neq k} x_i) x_k (x_1 + ... + x_k + ... + x_n) + (\Pi_{i \neq k} x_i) x_k = An^n (\Pi_{i \neq k} x_i) x_k + \frac{D^n}{n^n}$$

$$\frac{An^n}{D}(\Pi_{i \neq k} x_i) x_k^2 + \frac{An^n}{D}(\Pi_{i \neq k} x_i)(\Sigma_{i \neq k} x_i) x_k + (\Pi_{i \neq k} x_i) x_k = An^n (\Pi_{i \neq k} x_i) x_k + \frac{D^n}{n^n}$$

$$-An^n (\Pi_{i \neq k} x_i) x_k - \frac{D^n}{n^n} \quad - An^n (\Pi_{i \neq k} x_i) x_k - \frac{D^n}{n^n}$$

$$\frac{An^n}{D}(\Pi_{i \neq k} x_i) x_k^2 + \left((\Pi_{i \neq k} x_i)\left(\frac{An^n}{D}\Sigma_{i \neq k} x_i + 1 - An^n\right)\right) x_k + \frac{-D^n}{n^n} = 0$$

$$\div \frac{An^n}{D}(\Pi_{i \neq k} x_i) \quad \div \frac{An^n}{D}(\Pi_{i \neq k} x_i)$$

$$x_k^2 + \left(\Sigma_{i \neq k} x_i + \frac{D}{An^n} - D\right) x_k + \frac{-D^{n+1}}{An^{2n} \Pi_{i \neq k} x_i} = 0$$

$$x_k^2 + \left(\Sigma_{i \neq k} x_i - D(1 - \frac{1}{An^n})\right) x_k + \frac{-D^{n+1}}{An^{2n} \Pi_{i \neq k} x_i} = 0$$

IV.1.2. Analysis of roots and of derivatives

TODO

V. Bibliography

10 Egorov, Michael. 2019. "StableSwap - Efficient Mechanism for Stablecoin Liquidity." https://curve.fi/files/stableswap-paper.pdf.

Thomas, Morgan. 2021. "Danaswap: A Scalable Decentralized Exchange for the Cardano Blockchain." insert hyperlink here[1].

Wikimedia. 2021. "Newton's Algorithm." 2021. https://en.wikipedia.org/wiki/Newton's_method.

[1] https://insert%20hyperlink%20here