Report Update July-August 2023; Epistemology of Decoy Systems; Probing the Attacks on the Privacy of the Monero Blockchain

Nathan Borggren

CompDec

(Dated: September 1, 2023)

CONTENTS

I. INTRODUCTION

This document serves to report efforts and progress as recipient of a Magic Monero Fund [1]. As such it does not aim to be a completed scientific work, so much as to be a starting point for discussion, collaboration, future effort and a source of mathematical definitions for issues raised in [2] and [3].

In summary these issues correspond to information one party can glean about their counter-party through repeated transactions. The term *EAE Attack* or *Overseer attack* have been used. EAE stands for Eve-Alice-Eve, with the role of Eve usually being played by a government or an exchange. We would prefer to omit the words 'attack' and 'adversary' in favor of 'analysis' and 'counterparty' to adopt a more non-partisan stance. Eve is only acting in accordance with optimal play in a game theoretical sense, making due with all information available to her. Were this information gained by illicit acts, then 'attack' might be more instructive, but the information spoken of, connections and transaction values, can largely be gained through ordinary transactions. It is a point of confusion, at least for me, when the ends of a transaction are referred to by their moral proclivities rather than their name. In the intelligence community, 'attack' may refer to the attempt of money laundering where in the privacy oriented community 'attack' refers to the attempt of tracing the flow of funds. My concern here is with the actual fungibility of Monero, and am concerned with senders and receivers not attackers and victims. Through the inventions of tor (US Navy) and the establishment of security standards (NSA), we can see the sometimes bogey-men are also equally active and encouraging in developing protocols for privacy.

A. Global vs Local

A juxtaposition of scales occurs naturally in the Monero blockchain. Connections can occur locally, for example through direct interaction with a counter-party, or global, from the spending of coins minted as some particular block. Mesoscales, neither local or global are also created courtesy connections to decoys that occurred at intermediate scales. Scales in value as well as time occur, though this information is generally hidden it can be collected over time by actors that interact with numerous parties through numerous transactions.

Recovering hidden values or assigning hidden values is a typical task in quantitative finance. From evaluating the prices of IPOs to derivatives on stocks or other assets, or even more abstract notions like risk and liquidity, determining hidden values or getting bounds on hidden values is common place. In the *Value in the Monero Network* section some global approaches to determining value are discussed, one of them quantitatively. The EAE attack is generally local, even if the repeated transactions are over 6 months to a year, this still represents a small fraction of the total blockchain. Furthermore, not all transactions need to be explored, only a fraction of the total transactions will be present in a given taint tree. However, the information leaks of value are local, and the information propagate outwards constrains the expectation of value in other transactions. In [4] it was estimated that at the time $10-15\%$ of transactions involved ShapeShift as a counter-party. Thus ShapeShift or anyone substantially observant of the API for that time period has substantial information about the values of transactions.

The analysis for these global characteristics tends to be more computationally intense, but can be more straightforward to express mathematically. The attacks of interest in this study are more of the local varietal. A typical motif we'd analyze might be just a few transactions occurring over just minutes, hours, or days. A common and simple scenario is a small transaction, intended to verify receipt at an address, followed by the intended substantial transaction. We call this scenario the $2-AE$ motif and it already has the potential to leak some transaction history of the sender. Privacy-focused users may want to skip this verification step.

The information gained by an exchange, that is E or receiver in a $2-AE$ analysis is that the receiver can assume the sender has signing capabilities of the two transactions. They can also garnish which output belongs to the sender and follow it forward through successive transactions. E can then check for intersections between the ring constituents, looking for overlap of previous transactions. Efforts have begun to detect and quantify these overlaps, which do indeed occur; common histories can indeed be found. What remains to be shown is that any two transactions would also have these common histories generated through the decoy selection methods. It is interesting that the lack of asymptotic statistics, that each block has 10s to 100s of transactions rather than 1000s to 10000s, is actually helping matters, since it increases the likelihood these spurious connections, common histories that are not real common histories, do actually occur. We do also find overlap of taint trees of these random pairs of transactions, and are investigating further wether or not these share sufficient statistics to obfuscate real EAE patterns. In the *EAE experimental design section we set out on experiments to quantify this issue more succinctly.*

B. Statistical Attacks vs Deterministic Attacks

What we can gather about the tracing capabilities purported by Chainalysis and others on the Monero blockchain are of the statistical varietal. We speculate as to how these analysis may proceed and what disbelief must be suspended to believe these analysis. Almost surely no judge on our great planet will sit through an *almost surely* proof that there exists some possibility that the proposed transaction chain actually occurred and wait for this burden of proof to present itself. Warrants can be issued and subpoenas made on relatively sparse information. Property can be seized long before or entirely without a court decreeing to do so. We can imagine a range of responses along the Draconian spectrum ranging from tolerance to outright ban of Monero. On the laissez-faire end, law enforcement would rely entirely on the time stamps, the indelible truth of a transaction on Monero, and have to pull the thread of the actual humans/weapons/drugs being trafficked rather than the flow of cryptocurrency. Next would be guilt-by-association, which is similar to the logic of KYC laws already established. Herein interacting with scoundrels is tantamount to being a scoundrel oneself. Next would be guilt-by-bad-luck, where a party is considered a scoundrel by sharing a ring with a scoundrel. Finally, just guilt, you use Monero, ergo you are trying to hide your devious methods. We imagine, but don't know, that the United States is operating somewhere between guilt-by-association and guilt-by-bad-luck, as in if the probabilities are high enough, the federal jackets will sweep the floor. We can also imagine federal orders to Monero developers/miners that render it a violation of KYC to verify transactions over 10000 USD.

A retired NYPD officer, once upon a time implicated through spurious connections to the theft of the *Star Ruby* from the American Museum of Natural History confided with me, 'My innocence was besides the point. When all arrows point at you, all arrows are pointing at you.' There is thus the need to insure that the mixing and decoy selections that are occurring on the blockchain have the largest possible anonynimity set possible; rendering each transaction virtually indistinguishable with the other transactions that occurred at the same time.

This indistinguishability property is reminiscent of the early 20th century developments of Statistical Mechanics and ultimately Quantum Mechanics. Boltzmann inserted a $1/n!$ factor by hand to the partition functions in order to be consistent with the laws of thermodynamics. It took the introduction of Quantum Mechanics to explain what this factor was doing; accounting for the indistinguishability of the particles involved. No coloring of atoms or molecules was possible, one could never say 'it was this H_2O molecule not that one.' All H_20 molecules are effectively and actually the same, ie indistinguishable, the history of the trajectory of a molecule washed completely by thermodynamics and quantum mechanics. This level of indistinguishability should be a goal of Monero, currently transactions are like a red-dye propagating outwards, tainting it's path as it goes. We expect analogies from heat equations or fluid equations that quantify this mixing to be useful in the future, but we don't go down this pathway at this stage.

In the *Fitting Decoy Distribution* sections we measure some empirical distributions, we can then for any given ring look at all $n-1$ sized subrings to order the ring constituents in order of likelihood. We suspect the algorithms pushed as tracing to be of this varietal, and one merely chooses to believe the order of likelihoods the algorithm suggests, which could be sufficient to sell a product to a government or other Overseer, and issue warrants, regardless of the actual quality of the algorithm.

The EAE attacks are not of this varietal though, the connections an Overseer seeks are deterministic connections, demanding consistency between possible histories until only one true history remains. The random variables we use for transaction values also collapse to their deterministic variables, the counter-parties do indeed know the value of the transactions.

C. Hybrid Attacks

Hybrid Attacks would involve pursuing EAE determinism through statistical means. Namely sampling. We explore sampling methods as we were defeated when trying to exhaustively explore all paths. These sampling methods at this stage are sampled from uniform distributions, but we are developing the Bayesian update steps to explore the more likely transactions in a ring first. We also are developing are sampling methods to be exhaustive, removing paths as they arise so as to not be sampled twice.

D. Partial vs Complete Information

It is a goal for the privacy of Monero to be robust to small leakages of information, it should not matter globally if an exchange knows a few values and connections locally on the chain. Even large leaks where mass amounts of transaction information are present, should ideally be of negligible utility of transactions outside of that set.

E. Connectivity in the Monero Network

I can't speak for all parliaments across all nations and times, but we can suspect some common desires and choices with respect to the tracing of flows of funds across the Monero or any network. The fear from the government perspective is funds from illicit activity changes hands or funds change hands to finance illicit activity, their countermeasures evolved and are known as Anti-Money-Laundering (anti-money laundering). Obligations are placed upon exchanges to Know-Your-Customer, "know your customer" to prevent such matters. If a currency comes about that can clear transactions while bypassing these measures it is likely that legal measures will evolve to mitigate or prevent this. This process has begun in many jurisdictions. Currencies like this already exist, however, the dollar, the euro, the yuan etc. and this fungibility is generally considered a necessary condition on a Money.

However, with the advent of cryptocurrencies, opportunist surveillance industries took advantage of the lack of fungibility implicit to most blockchains to trace the flows of funds, so much so that they've come to expect this capability. Similar parallels exist for end-to-end private messaging with government reactions spanning the whole spectrum of tolerance to outright ban. Monero is also experiencing the same range of reactions across the planet. This effort here in no way promotes money laundering, indeed I discourage it. It does, however, seek to make improvements towards removing the historical traceability of cryptocurrencies to push it towards a more cash-like state. Just like the onus is on a cash-only bagel store to honestly report their earnings and pay taxes etc accordingly, the onus of a monero-only bagel store is to do the same. Whether or not they do so is not my concern nor the developers of cash, credit, or crypto.

At the same time I have no moral objection with a person, government, or an exchange to use all information legally available to them to get a clearer picture of the world around them and understand the interactions they are engaged in. In the end we have a classic evolutionary Red Queen scenario with all parties sharper as a result.

Pardon the interlude/disclaimer just some heat blowing on my neck.

Monero seeks to hide the sand at the beach, anonymity through obscurity, and does so by adding decoys to inputs to hide the true input. From a traceability perspective the lack of decoys in the outputs is problematic though. From a tracer's perspective every output is important, if it isn't the sender it is the receiver; both parties are of interest. In the case of churning, both parties are even the same, all paths forward are relevant and in some sense equivalent. From either the sender's or the receiver's perspective, the outputs are wholly de-anonymized; both parties know which output is theirs and which isn't. This fact is important in the context of the EAE attacks as it allows parties to build up a profile of their counterparty.

Perhaps an equally important issue with the large ratio of decoys/outputs is simply that there is an inefficiency present. More entropy, paths/kbyte on the blockchain, is available with more outputs. Let m be the number of decoys and transactions present at the input of a transaction and n be the number of outputs. The number of paths goes as $m * n$ whereas the space on the blockchain goes as $m + n$. For $m + n = C$ for some constant C, the maximum number of paths occurs when the number of inputs is equal (or a difference of one when C is odd). For a typical transaction with one ring input with 16 transactions and 2 outputs, C is thus 18, and the number of paths could be 81 rather than 32 for the same byte-cost on the blockchain. This could perhaps be implemented by generating multiple stealth addresses for either the sender/the receiver or both and splitting the corresponding outputs between those. This however ignores the issue that all outputs would still be of interest. It could be interesting to either use the additional outputs to pay for mining rewards rather than aggregating the mining rewards into a coinbase transaction or having 0 XMR transactions to ghost addresses. This could also have the added entropic benefit of some of the coinbase transactions appearing like any other transaction as the outputs get reused in the future.

In a previous work, correlations among the different rings of a multi-input transaction were shown. [5] This fact was purely statistical in nature, measured through counting, but it is possible that fears related to the EAE attack are already present at the multi-ring level. For example, for each pairwise combination between the two rings, run the taint tree backwards, just as you would investigating two transaction histories in the 2-AE attack.

We know that there is an enhancement in counts present when there is similarity between block heights, but it could be the case that not only are they the same height, but coming from the same transactions. That is to say, if the decoys are not effectively mixing then the histories of the *true* pair will overlap more than any other pair. Further efforts will explore if this is actually the case and if this statistical correlation can be rendered deterministic by deeper scrutiny of these pairwise taint trees.

This approach of course is rendered possible by the fact that there is one real transaction present in each ring. If there were rings entirely of decoys, or multiple real outputs in a single ring the correlations could be mitigated. Another approach could be to simply aggregate all the txs of all the rings into a single large ring, shuffle, and connect to the same outputs. With RingCT at 16, a transaction with two ring inputs has 256 possible pairs, whereas one RingCT of 32, two of which are real would have, 32 choose 2 or 496; nearly doubled. The situation is even more dramatic as the number of ring inputs increases. For the case of three ring inputs we'd have $\frac{(48 choose 3)}{16^3} \approx 4.22$, more than quadrupled the number of possibilities. A bonus benefit comes from a small drop in transaction bytes from the lack of a need of multiple ring hashes.

F. Value in the Monero Network

It is generally the case that tracers, like most folks, are more interested in large transactions than small ones. Although transaction values are obfuscated on the Monero blockchain there may be ways to recover some bounds. A few thoughts have occurred towards this end that I'll briefly discuss. One such avenue has quantitatively been explored.

1. Value through Optimal Transport

If you replace the word 'sand' with 'cons' and 'holes' with 'wallets' in [6] the rest follows. The classic picture in optimal transport is a pile of sand distributed over one region X, is moved into a distribution of sand over region Y. It takes some effort to move the sand from $x \in X$ to $y \in Y$, quantified by some cost c(x,y). A 'Plan' is some strategy, a probability measure in the product space, $\pi \in P(X, Y)$. This plan specifies exactly which sand in X goes to which hole in Y. Optimal transport then seeks to find the optimal plan; the one which minimizes the cost to execute.

For our situation with Monero, we need the relaxed, Kantorovich formulation (as opposed to the Monge formulation) since the coins can and generally are split.

Specifically, let X be the set of all coinbase transactions and let Y be the set of all utxos. Usually we would normalize to unit mass, though here it could be more natural to normalize to coins in circulation. The constraining equation $\int_Y d\pi(x,y) = d\mu(x)$ would simply be the coinbase value of the x transaction, read directly off of the blockchain. The equation is a fancy way of saying *The coinbase coins are now somewhere.* The complementary equation $\int_X d\pi(x,y) = d\nu(y)$ would then be the value corresponding to output y. It is a fancy way of saying *the coins in this output came from somewhere.*

A countable set of comparable equations can be created, constraining the number of plans we need to optimize over, by noticing this equation has to hold regardless of what time we look for utxos. For any block height we can consider the utxos as of that block height.

The cost used to evaluate a plan could be the probability, as measured by inverting the measured cdfs, to move from coinbase to the utxo. Some of these costs are infinite, indeed all costs outside of the taint tree for a transaction would be infinite. They have the interpretation that no coins from transaction y, could have come from transaction x. Similar infinite values will occur when we look at TDA through the filtration probability. Again it means that there is no transaction history present that can link the two transactions.

We do not explore this approach more at this stage, but we note that the sampling methods we develop are indeed sampling these types of plans.

2. Value through Derivative Pricing

In a 'risk neutral' framework the price of a derivative is simply the expectation value, the sum over all paths from present time to the expiry of the derivative, with each path weighted by the payout of that path times the likelihood of that path occurring[7]. Whereas it is the uncertainty of the future that sets the price of a stock derivative, it is the uncertainty of the past that sets the price of Monero in this analogy. This is to motivate the use of a stochastic variable in the place of the unknown value. We describe a preliminary approach to sampling this distribution, which will also relate to the distribution of the number of possible path histories for a given transaction.

Let us define a notion for 'implied paths,' a stochastic variable, for a given path to a coinbase sample. Notice these paths are also sampling the space described in the previous Optimal Transport section.

$$\#Implied\,Paths = \prod_{j=1}^{Max\,Depth} \frac{\#rings_j * \#mixins_j}{\#outputs_j} \tag{1}$$

Application of this equation and more discussion are included in the software section. A coupled set of equations is also used to describe value through these random variables.

$$tx\,value = \sum_{j=1}^{rings} ring\,value(j) \tag{2}$$

$$ring\,value = \sum_{j=1}^{decoys} tx\,value(j) * P(j) \tag{3}$$

Where $P(j)$ is the probability the jth transaction is the real transaction of the ring. Without additional knowledge this number is simply, $\frac{1}{\#\,decoys}$. As information is revealed, these probabilities could change, and even collapse to zero or 1.

The implied value of a tx from a single sample is simply $\#rings * coinbase\,value$

Although these formula only supply a stochastic look at the value of a given transaction, and thus do not achieve the deterministic goal we have for an EAE analysis, it is a belief of this author that these random variables when studied in bulk, can lead to some interesting measurements about the macroeconomics of Monero while maintaining privacy at the microeconomic level, which would be an achievement for the Monero developers. Also, as more gets known about the network, these distributions may end up getting tighter and tighter around particular values. Examples of such macroeconomic variables might be the effective money multipliers, average holding times, average transaction values, and with some additional assumptions, factorization methods (Principal Component Analysis (PCA) and Non-negative matrix factorization (NMF) the 'Mapper' algorithm often associated with (TDA) come to mind) might be able to find 'sectors' of the Monero economy.

3. Value through Linear Programming

Despite the vast number of unknown values for unknown transactions there are equally as many constraints on these values[8]. Furthermore, these constraints are linear.

The first constraint is that the sum of the values of the inputs is equal to the sum of the outputs (for the simplicity of notation we will consider the contribution to the miner's reward as an output.

$$\sum_{tx_i} v(tx_i) = \sum_{tx_o} v(tx_o) \tag{4}$$

The second constraint in it's most unassuming form is that the transaction value is greater than zero and less than the total number of coins in circulation. A much tighter constraint can be pulled from the taint tree. If we trace back the taint tree, every path originates as a coinbase transaction of some value. The upper bound then is merely the sum of all these coinbase values. This value would also be too large, as some paths exclude others yet all are counted, this number will still be much smaller than the total number of coins in circulation. Still we have an equation though for the constraint.

$$0 < v(tx) < \sum_{coinbase_i} v(coinbase_i) \tag{5}$$

We still would need a function to optimize over these constraints, which remains to be discovered, but the impulse is a functional that assigns a likelihood to each configuration of values based on the measured cdfs. As an estimate, pretending we have a hundred transactions in a block, and three million blocks, we are left with an unholy linear programming problem of 300 million unknown variables. Unholy, perhaps, but not entirely out of the realm of computational tractability. We'd also have 600 million constraints. These constraints are also incredibly sparse and might be deeply parallelizable, and are not dis-similar to Traveling Salesmen type problems an Amazon or Uber has to try to solve.

This framework could also be important in the Overseer context, since an exchange that has collected 1000s to millions of these transaction details, can naturally just adjust the constraints to include the additional information they have gleaned and potentially dramatically simplify the problem.

Check EAE Attack and Topological Data Analysis (TDA) RingCT

II. FITTING DECOY DISTRIBUTIONS

The obfuscation of the history of a transaction is a fascinating feature of the Monero blockchain. Every transaction is constructed with one or more rings and the real outputs are hidden amongst decoys. As a physicist, whose colleagues can tease out Higgs Bosons out of a slurry of particles, gravitational waves from the rest of the cosmic background, quantum coherence in a Faraday cage, the idea that one could hide a transaction among decoys, on a graph no-less, was an offensive one to me. Yet the decoy selection does seem to introduce enough Fear-Uncertainty-Doubt into a history to achieve the desired outcome of keeping the true history hidden. It certainly generates a mess while trying to explore and those smart-alecks who do use 300 inputs and 4000+ decoys in a transaction do successfully screech my brute-force approaches to a halt. However, my suspicions do remain, hence the methodologies conceived herein.

A few things are noteworthy of the implementation of the decoys.

- Transactions are held for 10 blocks before they can be reused.
- To account for changes in volume that do occur, a dynamic approach is used in selection for the recent transactions.
- By default, a Gamma Distribution, that has a very thin tail for both long and short times, renders a poor fit for recent times, and makes old transactions in rings rather surprising.
- Decoys are administered at the wallet level, not the protocol level, and multiple decoy selection algorithms have been deployed in the wild. Some even repeat entire rings, or otherwise trivialize the detection of the real transaction.
- the decoy selection improves with time, but heuristics noted from the past persist through some block range.
- Methods have gone from static to dynamic and efforts are being made to replace decoys with zero-knowledge proof setups

The details for which we are most concerned are the particular values for the probabilities associated with a given element of a given ring. We fit a gamma distribution to provide ourselves with a parameterized probability distribution we can subsequently call to determine the filtration parameter we will use in the Persistent Homology by Probability section. It has been pointed out to me that I used $log(block\,height)$ rather than $log(seconds)$, which could explain the deviation from expectations for the parameter results. This error provides a change of scale but not in change of ordering.

The resulting fits are shown for the alpha parameter in 4, 2, 3 below.

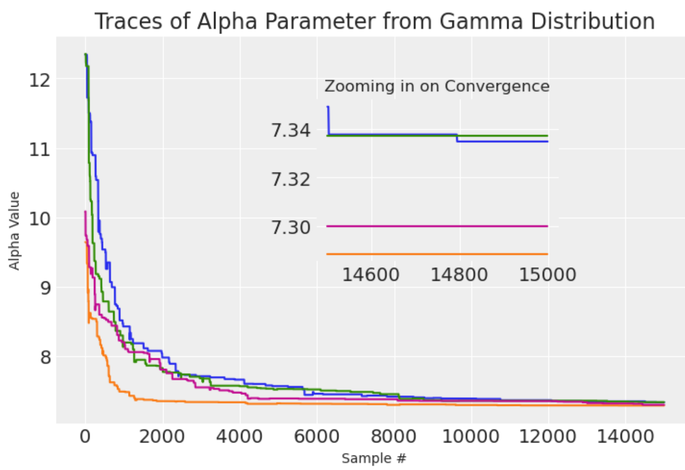

FIG. 1. Fits of the gamma parameter α. Inset zooms in on the region of convergence.

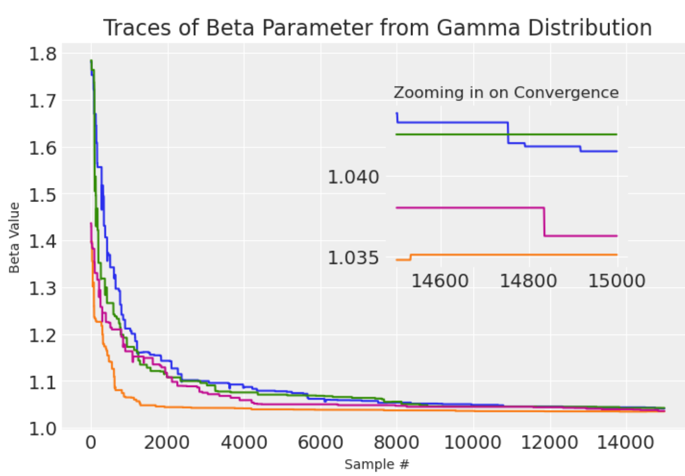

FIG. 2. Fits of the gamma parameter β. Inset zooms in on the region of convergence.

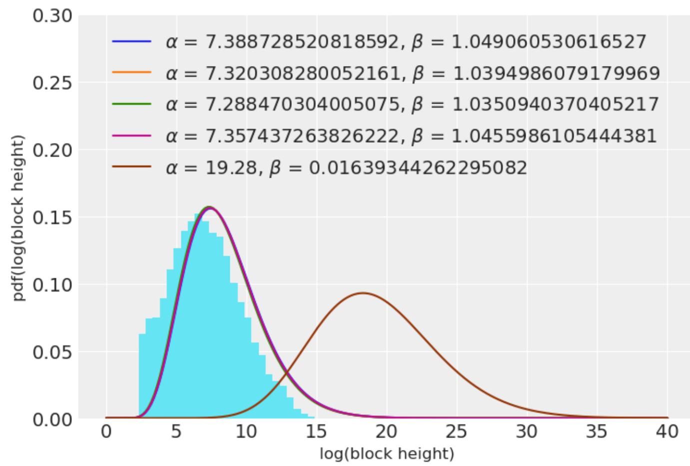

FIG. 3. The empirical, measured, and theoretical (erratum: wrong scale as described in text)

III. PERSISTENT HOMOLOGY OF A RINGCT

We will be using homology and persistent homology in multiple ways, the implementation and interpretation may differ between cases. In particular we will be using a *Vietoris-Rips Complex* to define the persistent homology for the RingCT case (where height is the filtration parameter) and a 'Flagser' method for the connectivity of the graph case. We will go into some detail showing the persistence diagram for a ring, which is the simplest case available as our metric space is 1 dimension (the block height).

The main idea of TDA is that of *persistence*; a sub-complex of a simplicial complex is constructed by providing a parameter and watching how that sub-complex changes from sub-complexes to the full simplicial complex as the parameter is swept. In our context, the transactions composing the ring are the vertices, the parameter being swept is the block height, and a vertex is joined with another vertex if its distance is within that height of the vertex. Persistent Homology uses the Union Find algorithm to find unions. In III we show which set each transaction in a ring is a member of as the algorithm progresses. Each vertex begins as the singleton set containing just that vertex

In practice we will simply call *Giotto's* Vietoris-Rips functionality and output a persistence diagram. Indeed this occurs when the *MoneroAna.tx* object is instantiated. We expect these block height persistence diagrams to be used in a multitude of ways.

- Unsupervised Machine Learning; the diagrams themselves occupy a metric space and can be used for clustering (bottle neck distances, Wasserstein distances, Frechet mean)

- Supervised Machine Learning; the decoy algorithm is implemented at the wallet level, not the protocol level, as such multiple decoy models exist in the wild. An experiment could be to generate transactions from a variety of wallets and develop a model to predict which wallet a signer of some transaction is using. In the context of EAE attacks, an exchange can potentially ascertain the external wallet used by a customer.

- Search optimization. This is my current focus and most relevant for the context of EAE. The intersection of taint trees can potentially be searched rather quickly by careful considerations of these diagrams. Say an exchange is looking for potential common transactions in the histories of two transactions. If the two transactions are the same then so too is the block height and so are the block ranges to all orders or persistence. Regions where the diagrams do not overlap can at least be temporarily ignored while regions of overlap are searched. I am looking for conditions (or reasons why they do not exist) in which the taint trees can be pruned and intersections can be found in potentially log(number of paths).

- Establishing Anonymity Sets/Confusion Matrices.

A. Worked example

For a given RingCT we'd like to be able to evaluate the likelihood subrings came from a decoy selection algorithm, find similar and comparable rings. We can also develop summary statistics about the nature of these rings and representations appropriate for Machine Learning.

In 4 we show the persistence diagram of a single ring. A log scale is shown to separate the points on the graph. A persistence diagram is a concise representation of all the information shown in Tables I, II, III.

Persistence diagrams are great at capturing structure at large scales. In I we see large scale structures; this guides the search to just bands of interest we can ignore or at least postpone queries for intersections in a large number of blocks when these diagrams are compared. We see as we zoom in at II, the structure reappear as the filtration parameter is reduced.

Usual histograms have washed away a lot of this information, and require choices of bin widths that this process can circumvent (or guide).

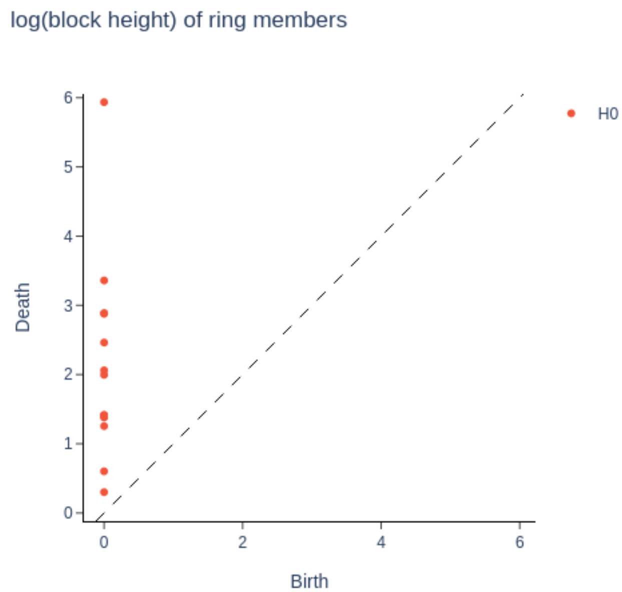

FIG. 4. Persistence diagram showing the birth-death pairs of a single ring. A log scale is shown to separate the points on the graph. A persistence diagram is a concise representation of all the information shown in Tables I, II, III. These diagrams can be analyzed in bulk to find means, anomalies, a basis for Machine Learning and More!

IV. PERSISTENT HOMOLOGY BY PROBABILITY

While persistence by height allows us to do some basic accounting and comparisons, it is not capturing the graph connectivity questions we are after. Nor does it allow us to explore the taint tree probabilistically. All transactions at a given time occupy the same set, they are not distinguishable one from the other. We introduce another construction that lets us try to connect with graph approaches to the analysis.

We will need a notion for distance, and we refer to the cdfs and fits computed in the fit section to do so. In the Ring object instantiation we require a set of decoys *and* a reference

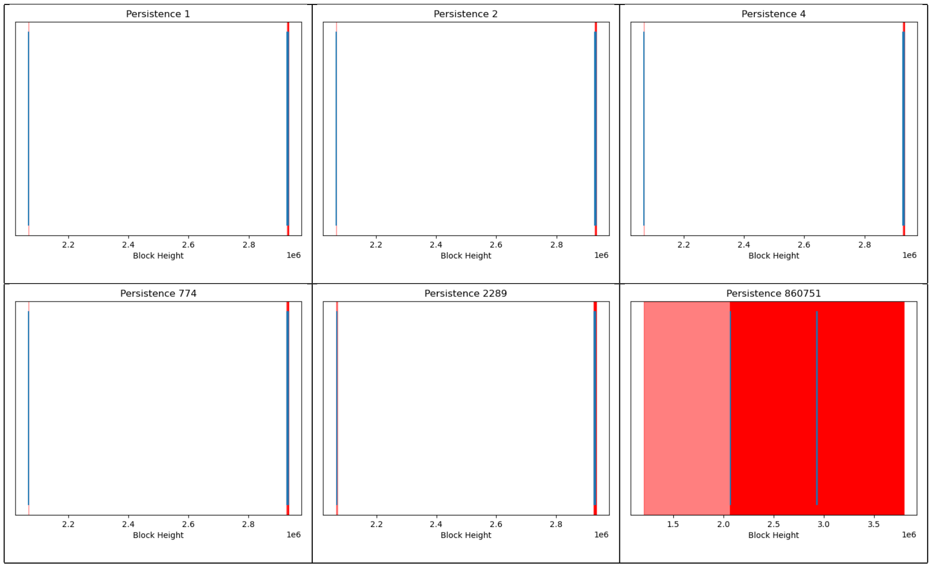

TABLE I. As the filtration progresses, holes are filled, joining neighboring transactions into a larger simplex. Only the first three and last three steps of the algorithm are shown, all of the structure during the intermediate heights is confined to the band on the right side, shown in greater detail in the next diagram. In this particular ring, one of the transactions is far older than the others, requiring a large parameter for the height to join the tx with the other transactions.

tx, we label txo for origin transaction. This allows us to do a few things. The ring needn't be required to actually exist somewhere on the blockchain, we can instantiate it with a different txo and place the ring in the context of a different txo. It is the case that rings have been re-used for different transactions [**?**], but they will still differ by different txo (must also differ in the real input too), and different hash.

These different txos change the offset, how long one must integrate to get the proper cdf, and thus the probabilities will be shifted monotonically as well. Furthermore, we can take as input a height persistogram, along with some parameter, to find a different tx that could be 'confused' with our tx (as in occupy the same simplex, and thus point to the same representative). This parameter when set to zero will force the sampled tx to have come from the same block as the target tx, and the probabilities will be identical.

The evaluations of the cdf in particular we are interested in are the integral of the pdf from time zero (the height of the txo) to the time of the height of the ring constituent. These give us the probabilities of the constituents being the real transaction. We can also consider the relative probabilities by normalizing; dividing by the sum of the evaluated cdfs. This has the more intuitive intrpretation of a weighted (currently) 16 sided dice.

We pivot to a distance notion by taking $1-q$ rather than q, so more likely things are the ones closer together, and certaicornties resolve on top of each other.

The registry objects, basically just a dictionary with keys the tx hash and values the tx object, can be used to construct the distance matrices we need to compute the homologies, or other graph metrics. We can recover spectra and other metrics for the corresponding graphs (1-skeletons) by setting the distance to one for each ring constituent.

A first attempt at constructing these matrices is included in the Taint-Explorer notebook.

A. Taint Trees

Persistence works a little bit differently than your intuition might have for probabilities. For example a path two-hops deep with .9 connecting the first and .9 connecting the second has probability of .81 of occurring, yet the two txs will already be connected when the filtration parameter reaches .9.

B. Sampling Paths

To sample paths each ring has a pymc categorical distribution over the RingCT that we can draw from. This distribution is also called in calls to the value of a ring or tx. We have considered all paths with equal opportunity at this stage. Fig. 5 shows a histogram of 3300 paths to coinbase from a transaction. We haven't parameterized this histogram at this stage, but we expect it to be exponential with mean related to the probability of drawing a coinbase transaction out of the ring, which terminates the sampling path. We can construct persistence diagrams for any of these paths, height paths are used to show the four diagrams in Table IV. For a given decoy selection algorithm, (or series, since this changes with block height), we can evaluate the likelihood of a given path to occur. Dynamic partition functions, that are weighted path integrals like this here, are called Maximum Caliber and have utility in statistical mechanics when the observables observed are not the energy paramater, but a categorical state (folded/unfolded, orbiting stationary points A,B,C etc.). These diagrams are used to estimate the value of a given tx, and to probabilistically sample the taint tree.

We can also look at a distribution of the values of the coinbases at 6. We expect taint trees of different txes with common true source to have comparable statistics. We need to check if transactions which could have been used interchangably as a decoy, also generate similar statistics. As mentioned in the Value section, these distrubitons of coinbases can be used to generate a probabilistic notion of value of an unknown tx.

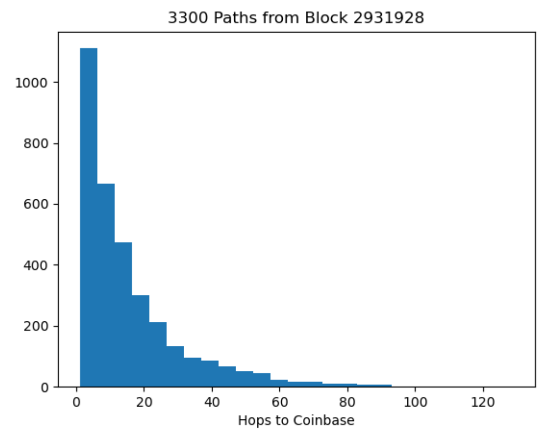

FIG. 5. A histogram of the length it takes to get to a coinbase, drawn from 3300 samples of a single transaction. These values can be used in the value expectation

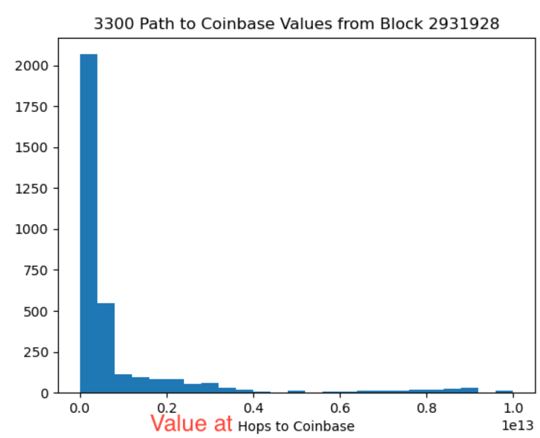

FIG. 6. A distribution of the values at coinbase of the separate paths. An estimate of the value of an unknown tx is the mean of this distribution times the number of inputs divided by the number of outputs.

V. EAE EXPERIMENTS

Although this section is very incomplete I'll describe the experiments that are underway. For the sake of this effort 15 transactions were made.

- 5 churning transactions from myself (monero-cli) to myself.
- 5 transactions from myself to a popular self-custodial wallet.
- 5 transactions from myself to an exchange.

These repeated transactions form the basis for our investigations of the 2-AE through 5-AE attack. The codes and results are still being verified, and I'll find a way to present the information in a redacted way for the sake of privacy. The preliminary results are that historical transactions can be found but spurious connections are also present.

We will be establishing experiments to find intersections of uncorrelated transactions to provide a background, ideally any false pair of transactions will also have intersections with similar statistics.

VI. MONEROANA

A git repository containing this documentation and of the python codes generated to produce the figures and results therein has been provided.

A. Basic Classes

Basic python classes were created to query and load the data as well as maintain close contact and syntax with the mathematics we will be using. As this analysis is primarily concerned with churning, EAE attacks, and other scenarios which can be characterized by involving relatively few actors and short time scales, the designs were made with composability and easy access in mind and to be used in a generative sense. For example $<, >, =, +, *$ are being overwritten so as to extend the functionality and convenience of the objects.

Other options were presented for the loading and interacting with the data and database or csv approaches might be of more use for more statistical analysis of the entire blockchain. The use case here is directed towards the user (or attacker) who is trying to understand the history and co-history of a potentially small set of transactions. The objects have a registry keyword that provides a context, basically a dictionary of what has been looked at already, whose keys are the hash and values are the objects instance in memory.

One can count on an adversary to have access to reasonable time and computing resources and willingness to spend hours, days, and months tracing the history of transactions. We therefore aim that any outcome of such a query results in maximal confusion with the maximal number of transactions.

It was a design choice, since the focus of this work is the local behavior in n-AE analysis, to keep a registry of every transaction visited over the course of a taint-tree exploration. This registry is a python dictionary with keys the hash of the tx, and the value a pointer to the instance of the Tx object described here. The tx objects maintains a list of inputs and outputs and appends to them as the tx arises in other contexts. The persistent homology by probability is implemented by providing a distance matrix directly and is the focus of the research. From these registries the relevant distances can be computed and the homology may commence.

10000 blocks is around two weeks of blockchain and all transactions therein held simultaneously in memory was manageable with a common laptop. When an instance of Block or Tx are created, a single query is made to an explorer and populated with the information therein. Maintaining the registry prevents the need for repeated calls to the api.

1. Block

The block object is instantiated given a block height.

- called with block height
- txs attribute provides list of tx hashes for the block
- *get_txs* attribute is a function that instantiates the Tx class for all the txs.
- obeys arithmetic properties using the block height as an integer. (in dev)

2. Tx

- instantiated with a call to the tx hash

- possesses attributes with the same names as the explorer api
- has a list of sources and a list of sinks maintaining a history of contexts the tx has arisen in
- *get_rings* instantiates ring objects for each ring input of the transaction.
- taint an iterator over the rings and mixins (in dev)
- value attribute, usually zero for non-coinbase transactions to be replaced with (pymc) random variable discussed in text. (in dev)
- required for taint tree sampling path computations

tx or transaction

3. Ring

A ring instance is called with a dictionary of inputs and a tx to serve as the parent node. Usually these are rings that have actually occurred on the blockchain, but we can do more. We can take the same ring of inputs and attach it to a different parent transaction,

- called with a collection of tx inputs and a txo, providing a parent node
- txs attribute provides list of tx hashes for the block
- *get_txs* attribute is a function that instantiates the Tx class for all the txs.
- obeys arithmetic properties using the block height of parent node as an integer. (in dev)

RingCT

B. Taint Trees, Sampling Paths, and Paths to Coinbase

Various functions have been created to enumerate and annotate the taint trees, sample paths up to a certain height, and create a bar code from the paths to coinbase as described in the text. The functions and documentation are in development.

VII. NOTATIONS

GLOSSARY

Anti-Money-Laundering: An envelope term for laws and regulations enacted to counter terrorism financing and money laundering.. 7

EAE Attack: The EAE Attack or Eve-Alice-Eve Attack. 12

Know-Your-Customer: Laws and regulations that require banking and other financial services to collect identifying information of customers using their service.. 7

RingCT: A signature formed with some number of decoy signatures as well as a real transaction. 12, 23

Topological Data Analysis (TDA): An applied mathematical discipline which seeks to analyze the shape of data. 12

tx or transaction: tx is used as shorthand for 'transaction' and as a variable name for the same. The subscript specifies the transaction with either a hash, or a variable or specific index to a hash like $tx_{e4ddaac1a449f3ec598b4cf30df1a86554...}$, tx_i, tx_5. 23

---

[1] Nathan Borggren. EAE Attack and Churning, 2023. URL `https://monerofund.org/projects/eae_attack_and_churning`.

[2] Ian Miers. Blockchain Privacy; Equal Parts Theory and Practice, 2023. URL `https://zfnd.org/blockchain-privacy-equal-parts-theory-and-practice/`.

[3] Monero Community Workgroup. Breaking Monero Episode 09: Poisoned Outputs (EAE Attack), 2019. URL `https://www.youtube.com/watch?v=iABIcsDJKyM`.

[4] N. Borggren, H.-Y. Kim, L. Yao, and G. Koplik. Simulated blockchains for machine learning traceability and transaction values in the Monero network, 2020.

[5] N. Borggren and L. Yao. Correlations of multi-input Monero transactions, 2020.

[6] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

[7] Paul Wilmott. *Paul Wilmott introduces quantitative finance.* John Wiley & Sons, 2007.

[8] James Burke. Linear Programming Review, 2023. URL `https://sites.math.washington.edu/~burke/crs/409/LP-rev/lp_rev_notes.pdf`.

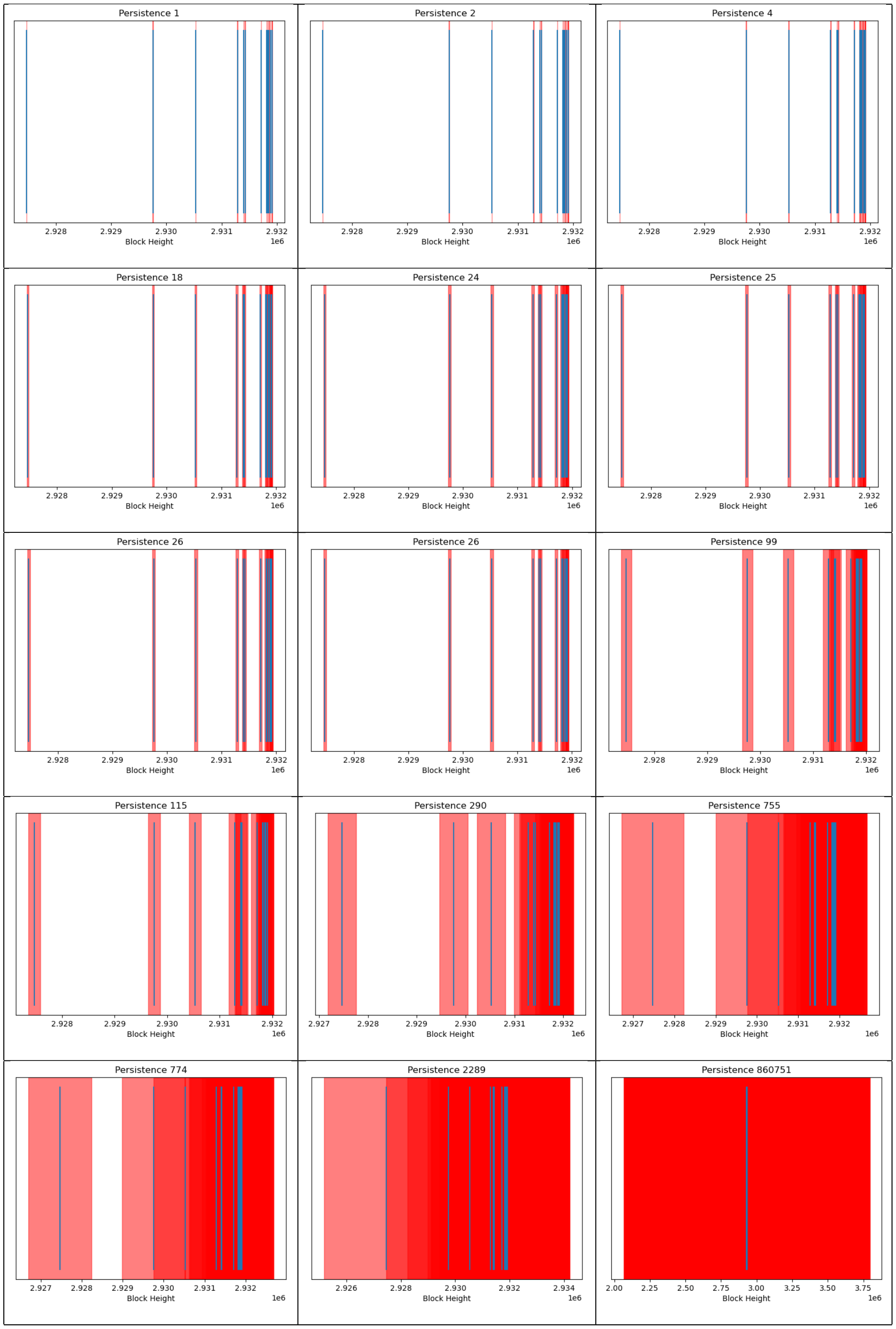

TABLE II. As the filtration progresses, holes are filled, joining neighboring transactions into a larger simplex. The fine structure at the different orders of the filtration are evident as we have zoomed into just the right side of the previous diagram.

block height	2066715	2927466	2929755	2930529	2931284	2931399	2931423	2931713	2931812	2931830	2931856	2931881	2931907	2931909	2931913	2931914
iter 0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
iter 1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	14
iter 2	0	1	2	3	4	5	6	7	8	9	10	11	12	12	14	14
iter 3*	0	1	2	3	4	5	6	7	8	9	10	11	12	12	14	14
iter 4	0	1	2	3	4	5	6	7	8	8	10	11	12	12	12	12
iter 5	0	1	2	3	4	5	5	7	8	8	10	11	12	12	12	12
iter 6	0	1	2	3	4	5	5	7	8	8	10	10	12	12	12	12
iter 7	0	1	2	3	4	5	5	7	8	8	10	10	12	12	12	12
iter 8	0	1	2	3	4	5	5	7	8	8	10	10	12	12	12	12
iter 9	0	1	2	3	4	5	5	7	7	7	10	10	10	10	10	10
iter 10	0	1	2	3	4	4	4	7	7	7	10	10	10	10	10	10
iter 11	0	1	2	3	4	4	4	7	7	7	7	7	7	7	7	7
iter 12	0	1	2	3	3	3	3	7	7	7	7	7	7	7	7	7
iter 13	0	1	2	2	2	2	2	7	7	7	7	7	7	7	7	7
iter 14	0	1	1	1	1	1	1	7	7	7	7	7	7	7	7	7
iter 15	0	0	0	0	0	0	0	7	7	7	7	7	7	7	7	7
iter 16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE III. Persistent Homology uses the Union Find algorithm to find unions. Here we show which set each transaction in a ring is a member of as the algorithm progresses.

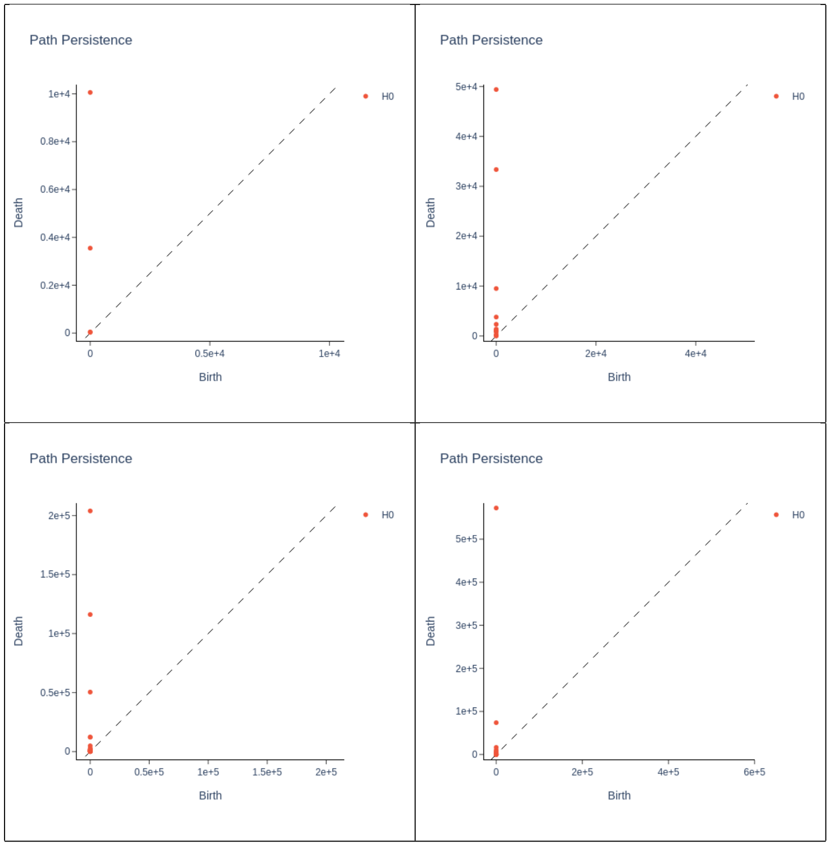

TABLE IV. Persistence diagrams of four sampled paths to coinbase. Diagrams with a few points have short trips to coinbase, diagrams with a lot of points have a lot of transactions prior to making it to coinbase. The spacings within the diagram specifies how large of block jumps were required to make it there.

Notations

tx	transaction identifier (hash)
tx_j	j-th transaction in set (often a ring)
$tx_{o,j}$	transaction output
$v(tx)$	transaction value
$r_j(tx)$	j-th ring input to tx
r_j	j-th ring input when particular tx is implied
$v(r_j(tx)))$	value of j-th ring input
$r_j, tx_k; r_l, tx_m; ...$	path identifier: the kth transaction of the jth ring
	followed by the mth transaction of the lth ring.
$\{$	start of a branching along a path
$\}$	end of a branch and return to parent node
$r_0, \{0; 2, 5; 1, 3\}\{1; 3, 1; 2, 4\}$	eg two paths out of the zeroth ring
	0th tx of r_0 followed by 5th tx of 2nd ring etc.