Initial Probability Density Function for Optimal Static Parametric Estimation of Arbitrary Distributions (OSPEAD)

Rucknium

January 15, 2025

Delivered to the OSPEAD Scientific Review Panel for Milestone 2 of the OSPEAD CCS proposal

Introduction

This document contains the best decoy selection distribution for Monero, based on data since the August 2022 hard fork. It also explains decisions made about the methodological issues that were discussed in the "Fully Specified Estimation Plan for OSPEAD" document submitted for Milestone 1. Details of step-by-step implementation and results are in the OSPEAD-docs website directory. See the README.md file contained in the top-level directory for more information.

Best Probability Distribution

The best-fitting decoy distribution was determined to be the log transformation of the generalized beta distribution of the second kind, with parameters scale = 20.62, shape1 = 4.462, shape2 = 0.5553, and shape3 = 7.957. On average, an adversary using the Maximum A Posteriori (MAP) Decoder Attack against rings constructed with this distribution would have correctly guessed the real spend in 7.6 percent of rings. This corresponds to an effective ring size of 13.2. Note that the minimum possible guessing probability is 1/16 = 6.25%. The estimated average attack success probability that an adversary can achieve against real Monero users who were using the default decoy selection algorithm since the August 2022 hard fork was 23.5 percent. This corresponds to an effective ring size of 4.2.

Revision, Deployment, and Disclosure

It is generally believed, though not proven, that deploying a new default decoy selection algorithm without a hard fork is worse than keeping a flawed one. An adversary could exploit the non-uniformity of the transactions on the chain to classify transactions and reduce anonymity pools to anonymity puddles. See my "Discussion Note: Formula for Accuracy of Guessing Monero Real Spends Using Fungibility Defects" for more discussion.[1]

Therefore, deployment of OSPEAD will likely wait until the next Monero hard fork. Yet, the next Monero hard fork will probably activate Full-Chain Membership Proofs (FCMP), which would eliminate rings altogether. Whither OSPEAD? OSPEAD is a good backup to FCMP in case its deployment is delayed unexpectedly and/or

[1] https://github.com/Rucknium/misc-research/blob/main/Monero-Fungibility-Defect-Classifier/pdf

major problems are found with its cryptography. In that case, a hard fork with a larger ring size could be deployed with the OSPEAD-derived decoy selection distribution. OSPEAD would be re-estimated with more recent data and feedback to the initial estimate in this document would be incorporated. The exact choice of distribution could depend not just on its raw defense number, 7.6 percent in the case of the generalized beta distribution of the second kind, but also possibly on the simplicity of its implementation in code.

I suggest that all OSPEAD-related documents and code be released publicly about a month from now, on February 20, 2025. Now that we have a complete method and proposed solution to the problem of timing-based attacks on ring signatures, it is time to open the method to more public scrutiny before possible deployment.

Role of OSPEAD After FCMP Activation

The original purpose of OSPEAD was to reduce the effectiveness of statistical attacks on ring signatures, which can be observed by an adversary with merely a copy of the Monero blockchain. Full-Chain Membership Proofs (FCMP) eliminates the need to select decoys for on-chain ring signatures, but privacy-preserving decoy selection can still have a role for users who require potentially adversarial remote nodes.

Monero wallets constructing FCMP transactions need to know their output's path in the FCMP Merkle tree. The current plan is to have wallets build the tree and keep a copy of it in their wallet cache as the primary method of getting their output paths. This method, though computationally expensive, prevents a remote node from learning anything about which outputs the wallet owns. However, the FCMP code will implement a backup method: query the node for the path of the output intended to be spent. Since the real output is requested in this method, decoy outputs can be request alongside the real output to obscure the real spend, just like with ring signatures. See Sections 6.9 and 6.10 of [Parker, 2024] and Monero dev meeting #82 affirming the use of this method.[2]

Given recent evidence that chain analysis firms use malicious remote nodes to reduce Monero user privacy, it continues to be important to protect users from statistical analysis of decoy requests [Kuyucu & Četyrkinas, 2024]. The FCMP decoy selection code can use the OSPEAD decoy distribution to maximize defense against possibly malicious remote nodes.

Remainder of the Document

The remainder of this document consists of comments about proposed methodological choices in the "Fully Specified Estimation Plan for OSPEAD" document. The are numbered by the section that they pertain to.

8 Modeling the Real Spend and Decoy Distributions as a Mixture

In lines 433-445 I discuss a small discrepancy with how the statistical model maps on Monero's protocol. Instead of each of the 16 ring members having an independent 1/16th probability of being selected from the real spend distribution, exactly one of the 16 ring members are from the real spend distribution. I stated, "I do not think that this theoretical problem causes a significant practical problem." As it turned out, it does cause a practical problem.

The consistency of the BJR estimator requires that the draws of each ring member be statistically independent of each other. Monero's ring members are not fully independent because exactly one ring member is always drawn from the real spend distribution. The rough criterion for determining if two random variables are independent from each other is to ask whether we get any information about the value of one random variable from the realized value of the other. Does this happen with Monero's ring signatures? Yes.

If one of the ring members has a value that suggests it is drawn from the real spend distribution, then that information can be used to guess that the other ring members will have been drawn from the decoy distribution.

[2] https://github.com/monero-project/meta/issues/1053

Think of the most extreme case: Rings with only two ring members, with the real spend distribution being a degenerate random variable always having a value of 0 and the decoy distribution always having a value of 1.[3] If the first ring member has a value of 1, we know that the second one will have a value of 0 always, and vice versa. These two random variables would have a correlation of -1 and therefore definitely would not be independent.

In practice, simulations suggest that the dependence in Monero's rings create a small but measurable inaccuracy in the BJR estimator under realistic conditions. See Chapter 3 "Successful Simulation" of OSPEAD-docs for a visualization of the magnitude of inaccuracy caused by the intra-ring dependence. The Kolmogorov–Smirnov statistic in that case was about 0.05.

I recent paper by Levine and Mazo suggests an estimator that can handle intra-ring dependence by using copulas[Levine & Mazo, 2024]. However, there is little hard proof that their estimator would produce accurate results. The [Levine & Mazo, 2024] estimator is based on the [Levine et al., 2011] estimator, which has never been proven to be a consistent estimator. Furthermore, [Levine & Mazo, 2024] acknowledge, "To the best of our knowledge, no identifiability results are available concerning this model." Therefore, I did not attempt to apply their estimator to the Monero ring signature data.

10 First Step: Bonhomme-Jochmans-Robin Estimator

Using the Jochmans' original MATLAB implementation as reference, I wrote a fast R implementation of the BJR estimator for the distribution component CDFs. At the request of isthmus, Jochmans gave the original implementation a BSD-2 license in November 2023, which removed any questions about adapting the code to an open-source implementation.[4] I also implemented an estimator of the mixing proportion suggested in the supplementary materials in the paper, which was not implemented in Jochmans' original MATALB codebase.

The original MATLAB implementation was unacceptably slow with ring size 16. Furthermore, it used a single CPU thread. The "BJR Benchmarks" appendix of OSPEAD-docs shows that my R implementation is about 240 times faster than the original MATLAB implementation when both are restricted to run on a single thread. My R implementation can take advantage of many CPU threads and even distribute the computations to many machines in a cluster arrangement. Nonetheless, it still took about two weeks to process two years of Monero blockchain data. The enclosed `decoyanalysis` R package contains the BJR implementation, callable as `bjr()`.

10.1 Linear Independence Assumption

The procedure to empirically check the linear independence assumption requires a test of a matrix's rank. As discussed later in "Determining the Number of Distribution Components K", There were unexpected challenges to finding a valid estimator of a matrix's rank. Therefore, the procedure to test this assumption was not implemented.

10.2 Permutation of Triples

Setting I to 10 performed well in simulations and was computationally feasible.

10.3 Determining the Number of Distribution Components K

Originally, the techniques in [Kasahara & Shimotsu, 2014] were going to be used to estimate the number of distribution components, i.e. the number of distinct decoy selection algorithms being used in the wild. The final step of [Kasahara & Shimotsu, 2014] required a statistical test of the rank of a matrix. After more research, I found that tests like [Kasahara & Shimotsu, 2014] would produce inaccurate results when statistical power is low. [Chen & Fang, 2019] discuss this issue.

[3] Thanks to isthmus for suggesting this explanation.

[4] https://jochmans.github.io/publications/melanges/JRSSB%20Replication%20Material.zip

In simulations, the BJR estimator performs well when the K is specified to be larger than it actually is. The "phantom" components have an estimated mixing proportion near zero. See Chapter 3 "Successful Simulation" of OSPEAD-docs for the results of one such simulation. Given questions about the accuracy of the [Kasahara & Shimotsu, 2014] estimator and the good performance of the BJR estimator when specifying a high K, I decided to set K to 4 for all weeks because the number of unique decoy selection algorithms is unlikely to exceed 4. In the actual estimation results for real data, the estimated mixing proportion of the fourth distribution component was close to zero, suggesting that the number of unique decoy selection algorithms may not have exceeded 3.

11 Second Step: Patra-Sen Inversion Estimator

The Patra-Sen inversion estimator was implemented in the `decoyanalysis` R package as `patra.sen.bjr()` .

12 Confidence Interval Estimation

I originally suggested a bootstrapping procedure to estimate confidence intervals. Given that a single round of BJR estimates of two years of Monero data takes about two weeks to complete, repeating the estimation 100 or more times for bootstrapped confidence intervals is not feasible.

13.2 MyMonero Fee Fingerprinting

To identify on-chain transactions as constructed by MyMonero would have required a deep analysis of MyMonero's fee algorithm. An attempt to have a C++ programmer to help demystify the MyMonero fee algorithm was unsuccessful.

13.3 Nonstandard Wallets are Nonstandard in Multiple Ways

Transactions that could be identified as nonstandard were removed from the dataset before estimation. See Chapter 5 "Nonstandard Transactions" in OSPEAD-docs.

An unexpected source of nonstandardness appeared when the off-by-one-block decoy selection bug was patched.[5] After the patch, two wallet2 decoy selection algorithms were being used in the wild. I attempted to estimate the weekly proportion of transactions that were constructed with the old and new wallet2 decoy selection algorithms, using the techniques in [Hall, 1981].. However, the results were unsatisfactory because the difference between the two distributions was so small. In the end, I assumed an adoption curve. See OSPEAD-docs for implementation details.

18 OSPEAD Requests to Monero C++ Developer(s)

Item (1), "Converting the `wallet2` decoy selection algorithm into a closed-form probability density function, i.e. $f_{wallet2,D}$" was accomplished by jeffro256.[6] Items (2)-(5) were not done.

21 Criteria for Best Fit

Chapter 15 "Rucknium Ratio Attack/MAP Decoder/Discriminant Analysis" discusses the history of the MAP decoder attack against ring signatures. In my opinion, the long analytical history strongly suggests that the MAP decoder attack is the optimal timing-based attack against ring signatures. The best defense against the attack is to choose a distribution that minimizes the attack effectiveness. Therefore, I chose to use the attack success

[5] https://www.getmonero.org/2023/06/08/10block-old-decoy-selection-bug.html
[6] https://github.com/monero-project/monero/pull/9024

minimization as the single criterion for best fit instead of using each of the six distinct criteria discussed in this section.

Equation 19 specifies a weighting function for outputs, parameterized by λ. The weighting function can give more or less weight to older outputs. Originally, I suggested trying $\boldsymbol{\lambda} = \{0, 0.5, 0.9, 0.95, 0.99, 0.999, 0.9999, 1\}$ as possible values for λ, thinking that small deviations from $\lambda = 1$ would cause large changes in the fitted distributions. As it turned out, small deviations had almost no effect on the fitted distributions. Therefore, only $\lambda = 1$ and $\lambda = 0.5$ were tried.

In choosing the parametric distributions to fit, I aimed to use general distributions that include other distributions as special cases. Thus, it is as if we are fitting a large number of different distributions in a single optimization procedure. These distributions were fit:

- **Generalized Extreme Value (GEV)**. Includes Gumbel, Fréchet and Weibull (also known as type I, II and III extreme value) distributions as special cases [Coles, 2001].
- **Generalized Beta of the Second Kind (GB2)**. Many distributions are a special case of this distribution. These special cases are the Burr (Singh-Maddala), Dagum, beta distribution of the second kind, Fisk (log-logistic), Lomax (Pareto type II), inverse Lomax, generalized gamma, gamma, and Weibull [Kleiber & Kotz, 2003].
- **Gamma**. The exponential and chi-squared distributions are special cases of the gamma distribution [Kleiber & Kotz, 2003].
- **Non-Central F**. The central F distribution is a special case.
- **Right-Pareto Lognormal (RPLN)**. Used for modeling size distributions [Reed & Jorgensen, 2004].
- **Birnbaum-Saunders (BS)**. Used for modeling metal fatigue, cracking, and failure times [Birnbaum & Saunders, 1969].

Notice that the gamma distribution is a special case of the GB2 distribution. I decided to fit this special case separately since it is useful to compare with Monero's status quo Log-gamma decoy distribution.

24 Dynamic Risk and Forecasting

After FCMP is activated, there will be no more on-chain data available to estimate the real spend age distribution. All that will be available will be historical data. Therefore, neither an adversary nor Monero developers can adjust their algorithms based on updated data. Given that OSPEAD is expected to be deployed in a post-FCMP environment, I decided that forecasting was an unnecessary step. Furthermore, the preliminary test results in Section 25.3 "Evaluation of Forecast Accuracy" using transparent coins' data did not show much improvement when using a complicated forecasting algorithm compared to a naive average of the past distributions.

Acknowledgments

This research was funded by Monero's Community Crowdfunding System:

https://ccs.getmonero.org/proposals/Rucknium-OSPEAD-Fortifying-Monero-Against-Statistical-Attack.html

Thanks to isthmus (Mitchell P. Krawiec-Thayer) for discussion of certain methodological issues and for creating a nonstandard transactions list. Thanks to jeffro256 for improving the documentation of Monero's default decoy selection algorithm. Thanks to plowsof for lending computing resources for collecting txpool data. Thanks to gingeropolous for managing the Monero Research Lab Research Computing Cluster, which was used to perform the computations for the statistical estimates.

The following people gave feedback on the research process and/or contributed in other ways: ACK-J, ArticMine, bob, coinstudent2048, garth, hyc, jberman, kayabaNerve, koe, mj-xmr, monerobull, moneromooo, neptune, SamsungGalaxyPlayer, SerHack, SethForPrivacy, and Syksy.

References

[Birnbaum & Saunders, 1969] Birnbaum, Z. & Saunders, S. (1969). A new family of life distributions. *Journal of Applied Probability*, 6(2), 319–327. https://doi.org/10.2307/3212003

[Chen & Fang, 2019] Chen, Q. & Fang, Z. (2019). Improved inference on the rank of a matrix. *Quantitative Economics*, 10(4), 1787–1824. https://doi.org/https://doi.org/10.3982/QE1139

[Coles, 2001] Coles, S. (2001). *An Introduction to Statistical Modeling of Extreme Values*. Springer London. https://doi.org/10.1007/978-1-4471-3675-0

[Hall, 1981] Hall, P. (1981). On the non-parametric estimation of mixture proportions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 43(2), 147–156. https://doi.org/https://doi.org/10.1111/j.2517-6161.1981.tb01164.x

[Kasahara & Shimotsu, 2014] Kasahara, H. & Shimotsu, K. (2014). Non-parametric identification and estimation of the number of components in multivariate mixtures. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 76(1), 97–111. http://www.jstor.org/stable/24772747

[Kleiber & Kotz, 2003] Kleiber, C. & Kotz, S. (2003). *Statistical Size Distributions in Economics and Actuarial Sciences*. Wiley. https://doi.org/10.1002/0471457175

[Kuyucu & Četyrkinas, 2024] Kuyucu, İ. & Četyrkinas, L. (2024). *How chainalysis made their way into popular monero wallets*. https://www.digilol.net/blog/chainanalysis-malicious-xmr.html

[Levine et al., 2011] Levine, M., Hunter, D. R., & Chauveau, D. (2011). Maximum smoothed likelihood for multivariate mixtures. *Biometrika*, 98(2), 403–416. https://doi.org/10.1093/biomet/asq079

[Levine & Mazo, 2024] Levine, M. & Mazo, G. (2024). A smoothed semiparametric likelihood for estimation of nonparametric finite mixture models with a copula-based dependence structure. *Computational Statistics*, 39(4), 1825–1846. https://doi.org/10.1007/s00180-024-01483-4

[Parker, 2024] Parker, L. (2024). *Fcmp++*. https://github.com/kayabaNerve/fcmp-plus-plus-paper

[Reed & Jorgensen, 2004] Reed, W. J. & Jorgensen, M. (2004). The double pareto-lognormal distribution—a new parametric model for size distributions. *Communications in Statistics - Theory and Methods*, 33(8), 1733–1753. https://doi.org/10.1081/STA-120037438