Extreme DARK: A more secure DARK variant

Michael Brand

November 23, 2023

Executive Summary

The FinTracer DARK algorithms are an extension of the FinTracer algorithm designed to enable creation of FinTracer propagation graphs even where no shared transaction data is available. In the Extreme DARK algorithm an honest-but-curious Financial Intelligence Unit (FIU) receives less information about general statistics regarding the DARK signatures used, and the information nevertheless revealed is better protected in a differential privacy sense.

1 Introduction

The FinTracer algorithm, which has become the algorithmic backbone behind much of the Purgles Platform, is a powerful and versatile algorithm that utilises the fact that in financial transactions both reporting entities (REs) involved in any transaction are aware of its details. If we consider the financial system as a transaction graph, FinTracer utilises the facts that the edges of this graph are visible to both involved REs to allow searching and matching of interesting typologies across the entire graph. This is done by propagating information along the transaction graph, which acts here as a *propagation graph*.

The problem, however, is that the transaction graph is not the only graph that is useful, from an intel perspective, to use as a propagation graph, and other graphs of practical interest do not have this property.

FinTracer DARK is an extension of the FinTracer algorithm that is meant to enable creation of FinTracer propagation graphs even where no shared transaction data is available.

The DARK algorithms have one major concern in that the algorithm relies on the existence of cryptographic keys known to all REs but not to the FIU.

The present document is a further evolution of the DARK family of algorithms. It abolishes the shared private key, however the algorithm's security does not not protect against information leakage in the event of collusion between the FIU and any RE.

The algorithm does, however, minimise the leakage of aggregate, statistical data to an honest-but-curious FIU.

In Section 2, we briefly recap the DARK algorithm. In Section 3, we describe the new algorithm variant. In Section 4, we show how to choose run parameters for the new algorithm. And in Section 5, we conclude by analysing the new algorithm's costs and its security implications.

2 Algorithm recap

The basic premise of FinTracer DARK is as follows.

A FinTracer *tag* is a mapping from accounts to *tag values*, each of which is an ElGamal semi-homomorphically encrypted value, encrypted with a key that only the FIU has access to. The main step of the algorithm involves propagating these tag values along a transaction graph. In DARK, the propagation must be done in a situation where the tag-sending party does not know who the receiving party is. Indeed, the algorithm must even protect it from knowing how many recipients each tag had, even though some tags may be received by all reporting entities (REs) and forwarded to many accounts in each RE, whereas others are not received at all.

Similarly, receiving parties must not know whether they are receiving tags from many accounts at every other RE, or whether they are not receiving tags at all.

For this, the algorithm defines *DARK signatures*. These can be thought of as *mailboxes* that reside at the midpoints of propagation graph edges. Instead of having the REs send tags to each other, as they do in FinTracer, in DARK the REs send the tag values to mailboxes, and the values then get read by the receiving RE which distributes the tag values to the correct accounts.

Thus, each edge now becomes two half-edges and a mailbox. In fact, both on the sending side and on the receiving side the relationship between accounts and mailboxes is many-to-many: many accounts may share the same mailbox, and any account may use many mailboxes.

For example, a mailbox may indicate the account owner's name, and all accounts whose owner is "John Smith" will read and write the same mailbox. Furthermore, in order to allow fuzzy matching, we may also have them populate the mailbox labelled "Johnny Smith". Even without fuzzy matching, an account may map to more than one mailbox, e.g. because it has multiple owners.

Many of the algorithm's parameters depend on the maximum number of mailboxes any particular account can use. Let T_r be the maximum number of mailboxes an account can read, and T_w be the maximum number that it can send to.

The way that FinTracer DARK works is that the REs coordinate between each other a private/public key pair, as well as a hashing key, so that all REs know the private key and the hashing key, and the FIU only knows the public key. The public key is used to encrypt the tag values to be sent and the hashing key is used to encrypt the ID of the mailboxes. Once that is done, the doubly-encrypted values and their associated singly-encrypted destination addresses are sent to the FIU.

Meanwhile, the receiving REs send to the FIU the encrypted IDs of the mailboxes they wish to read from. The FIU relays each tag value to its designated mailbox, sending the final mailbox value to the REs that have registered as recipients of each mailbox's value.

Importantly, receiving REs should not know whether any particular mailbox was populated with a value from a sending RE. The FIU, using the public key that it holds, sets the mailbox value of any "empty" mailbox to an encrypted zero.

Note 2.1. AUSTRAC delivers to each mailbox $Rvalues, where$R is the number of participating REs. The reason for this is that the values received from each RE are encrypted, so it was thought AUSTRAC would not be able to sum them up. In fact, the second layer of encryption applied to tag values can also be ElGamal encryption, in which case we can use the tools of [4] to allow the FIU to sum values even through the double encryption, so that each mailbox receives only a single value.

3 The new algorithm

The key difference between the algorithm described in Section 2 and the present algorithm, Extreme DARK, is that the original algorithm was reliant on the existence of a single hashing key and a single private key shared between all REs.

In Extreme DARK, we instead require each of the $R(R-1)$ communicating (ordered) pairs to agree on their own hashing key. Also, instead of a single global private key, each RE has their own private key that is used to encrypt any tag value that is sent to them. Thus, instead of 2 keys used in the system (not counting the FIU's key), the new algorithm requires R^2 ones.

Additional follow-on differences are:

1. Instead of communicating its tag data and mailbox IDs only once to the FIU, each party must repeat the process $R-1$ times, each time using a different key;

2. The trick of Note 2.1 no longer works. The FIU will need to deliver the $R-1$ values separately, for the recipient to sum them up; and

3. The computation of how many fake signatures are needed is different. (See Section 4 for details.)

Note 3.1. In the previous algorithm DARKer, we speak of R tags being delivered. Here: $R-1$. The difference is merely an implementation choice: we assume here that all communication from an RE to itself does not go through the entire process described. Instead, each RE's communication to itself is handled locally.

Supporting such optimisation requires the implementation of minor additional complications in the relevant FTIL scripts, but in the case of Extreme DARK the benefits substantially outweigh these complications.

4 Fake signature generation

4.1 Background

We will here provide an explicit analysis regarding the the number and manner of fake signatures (fake mailbox IDs) required for the present algorithm, noting that it slightly simplifies the problem, and also that it will, in general, require less fake signatures.

Before going into the details, however, we recap the theory behind fake signatures.

When using FinTracer DARK, the FIU has no visibility towards the values of the tags or the signatures being communicated. It also cannot see how many accounts of any given RE use a particular signature. It does, however, see the number of distinct signatures produced by each individual RE.

Exteme DARK mitigates the issue of it being possible for the FIU to see regarding each signature which REs use it to either send or receive tags. Now the FIU only has visibility towards three types of knowledge:

1. How many signatures, altogether, does each RE send to?
2. How many signatures, altogether, does each RE receive from?
3. For each pair of REs, how many of their signatures match?

These numbers cannot be completely hidden, but they can be obscured by the introduction of "fake" signatures. Specifically, we create different fake signatures to obscure each of the three types of knowledge listed above, i.e. some fake signatures serve only as fake mailbox IDs to send to, some as fake mailbox IDs to read from, and some as fake mailbox IDs that are both written to and read from. Where tags are written to fake mailboxes, the tag values are random.

To determine how many fake signatures of each kind are required, we use the criterion of (ϵ, δ)-differential privacy. This criterion, explained in detail in [2], essentially requires that for each element whose details we wish to keep private (in our case, any account) it is required that adding or removing the element will not change the probability of any possible visible output of the algorithm, except by at most a multiplicative factor of e^{ϵ}, and except for a set of outputs whose total probability is at most δ, where e is Euler's constant. Importantly, this criterion must hold for any choice of element relative to any background set (i.e., in the worst case). All probabilities are measured based solely on the random choices of the algorithm, not based on the choice of element or background set.

4.2 Generation algorithm

Let $AEXP(n, \gamma)$ be the distribution over the integers in the range $[0, \infty)$ such that the probability of a random variable allotted from this distribution to equal x is proportional to $e^{-\gamma|n-x|}$.

We refer to this distribution also as the (ϵ', δ')-AEXP, reparameterising it by using $\gamma = \epsilon'$ and

$$n = \max\left(0, \left\lceil \frac{1}{\epsilon'} \log\left(\frac{1-e^{-\epsilon'}}{\delta'}\right)\right\rceil\right).$$

As was proved in [3], this distribution satisfies that if the random variable $\mathbf{x}$ is distributed (ϵ', δ')-AEXP, then

$$\text{Prob}[\mathbf{x} = 0] \leq \delta'.$$

This, in conjunction with the fact that a change of $\mathbf{x}$ by 1 alters the probability of $\mathbf{x}$ by a multiplicative factor not exceeding $e^{\epsilon'}$, and the fact (not proven here) that this is, essentially, the distribution with minimal expectation over the same support that satisfies these criteria, makes it ideal for our purposes. (As an added bonus: as demonstrated in [1], it is also easily implementable in FTIL.)

Using the AEXP distribution, we define our signature generation algorithm as follows.

1. Let $\mathcal{A}$ be the $\left(\frac{\epsilon}{(T_r+T_w)(R-1)}, \frac{\delta}{(T_r+T_w)(R-1)}\right)$-AEXP distribution.
2. Each RE r_1 generates for every other RE, r_2, a number $m_{r_1 r_2} \sim \mathcal{A}$.
3. r_1 sends $m_{r_1 r_2}$ to r_2.
4. Each RE r_1 generates $f^r_{r_1}, f^w_{r_1} \sim \mathcal{A}$.
5. When r_1 is sending to r_2, both REs generate $m_{r_1 r_2}$ fake matching signatures (e.g., by following a pattern such as "Fake signature #n").
6. The total number of fake signatures generated by r_1 when sending to r_2, including both matching and non-matching ones, is $\max_r (m_{r_1 r}) + f^w_{r_1}$, where the additional signatures are random (meaning they will not be matched).
7. The total number of fake signatures generated by r_2 when receiving from r_1, including both matching and non-matching ones, is $\max_r (m_{r r_2}) + f^r_{r_2}$. (These will also not be matched.)

Note 4.1. By ensuring that the total number of mailbox IDs written to by r_1 is the same for each r_2, and that the total number of mailbox IDs read from by r_2 is the same for each r_1, no unnecessary information is provided to the FIU.

To explain the use of the specific AEXP parameters in the distribution $\mathcal{A}$, consider the following.

First, suppose that due to some change at RE r_1, r_1 now wants to write to one additional mailbox. (The example where r_1 wants to write to one mailbox *fewer* is symmetric.) When communicating to r_2, that additional mailbox may either be matched (causing an increase of 1 in the number of matched signatures) or unmatched (causing an increase of 1 in the number of unmatched signatures). In either case, this is a minimal-sized change in the values that the FIU can track simply by being honest-but-curious.

Suppose that we provide (ϵ', δ')-differential privacy protection for such a change.

Over the course of the entire protocol, r_1 communicates not just to r_2, but to every one of the other REs, and if r_1 requires an additional mailbox, this will cause a "minimal-sized change" such as the one described above in the number of signatures used by r_1 in communicating to each of the $R-1$ other REs. The overall privacy protection afforded by the algorithm against such a change is therefore at least $(\epsilon'(R-1), \delta'(R-1))$-differential privacy, because it can be viewed as a protection against $R-1$ changes.

An equivalent situation appears also in the case where r_1 changes the set of mailboxes it reads from.

The elements being kept private by the differential privacy constraints of the algorithm are, however, not the signatures themselves. They are the underlying accounts. If r_1 opens one new account, that account may write to T_w additional mailboxes and may read from T_r additional mailboxes.

In total, the algorithm provides $(\epsilon'(R-1)(T_r+T_w), \delta'(R-1)(T_r+T_w))$-differential privacy in protecting the identity of the accounts at the various REs.

The choice of parameters for $\mathcal{A}$ ensures that this total privacy protection is (ϵ, δ)-differential privacy, as desired.

5 Analysis

5.1 Security analysis

FinTracer DARK largely follows the FinTracer framework, which makes it, in many ways, have the same cryptographic strengths and weaknesses.

The main difference between the DARK family and classic FinTracer, security-wise, is in DARK's increased vulnerability if the FIU does not follow the protocol. In classic FinTracer, the FIU plays no role during the propagation of tags, making the process maximally secure against information leaks towards an FIU that doesn't follow the protocol, by minimising its exposure to data. For the FIU to gain illicit information during the tag propagation process, it must necessarily collude with one or more of the REs. Even so, any illicit information gained in this way is limited by being information that is in the possession of the colluding REs. Tags that are propagated between two REs neither of which is in the colluding party remain invisible to the FIU.

In the DARK family, almost by definition, this is not the case, and Extreme DARK is no exception.

In DARK, the FIU acts as a postmaster, delivering the messages between the REs. The individual REs sending tags do not know where these tags will end up, and the individual REs receiving tags do not know where these tags came from.

Thus, the FIU is no longer kept physically separated from any of the tags being propagated in the system.

Consider, now, a situation in which the postmaster colludes with another RE, r. Any RE sending any tag whatsoever in the system does not know whether this tag should or should not be routed to r. If the FIU colludes with r, the FIU can deliver the tag to r in every case, so the content of all tags in the system is visible to the colluding parties.

For the same reason, a colluding r must be able to determine which signatures the other REs are sending to and—considering tags sent by r—also which signatures they are receiving from.

These signatures themselves are hashes, but given the collusion the hashes are no longer cryptographically protected. They are subject to a dictionary attack: RE r can try out every signature it knows; if the space of signatures is small it can exhaustively try out every signature, and if it is large it can work from the most likely signatures down. For example, if the signatures represent the names of account owners, r can simply try out every name in a telephone directory to reverse the hash of signatures.

Thus, in all DARK variants privacy guarantees are not robust to collusion, specifically collusion between the postmaster and one or more REs.

Extreme DARK, however, performs better than other DARK variants in the case of an FIU that is non-colluding, but is honest-but-curious or otherwise does not follow the protocol.

In the DARKer algorithm variant, for any subset S and subset T of the REs, an honest-but-curious FIU would have known how many signatures are sent by exactly the members of S and received by exactly the members of T. This is an exposure to a total of 4^R values, where R is the number of participating REs. These values can be obscured using differential privacy, but this requires a much larger number of fake signatures, and the approximate numbers themselves are already quite revealing.

By contrast, as discussed in Section 4, in Extreme DARK the FIU is only privy to (approximate versions of) $R^2 + R$ numbers.

On the other hand, Extreme DARK has an added difference, compared to the previous DARK algorithm, in that REs could choose to not follow the protocol in novel ways: instead of sending the same signatures and values to all REs, they can send distinct ones to each RE. This is a special case of the REs choosing not to use their real data, and does not directly cause an information leak. It may also be regarded as a feature: where REs know which party or parties they mean to send to, they can send only to those.

5.2 Extreme as FinTracer: A cost comparison

Where the security differences listed above are not a concern, the new protocol can even be used as a replacement for standard FinTracer even when the propagation graph is simply a transaction graph.

This is useful, because it has the advantage that pairs of REs do not need to synchronise their transactions in the clear before the start of tag propagation - and there may be situations where it is preferable to avoid this synchronisation.

Most importantly, in DARK the Coordinator Node, sitting at the FIU, is a single communications hub, with all communication having to go through it, whereas FinTracer is a fully distributed system, with most data flowing between peer nodes without any mediation by the Coordinator. This is in many contexts an advantage for FinTracer, because the Coordinator Node does not become a bottleneck.

However, if for any reason REs determine that it is not feasible for them to communicate with each other directly, Extreme DARK provides a solution.

Thus, there is a sense in which Extreme DARK can be thought of as (in addition to everything else) also a replacement for FinTracer. In such a perspective, it makes sense to compare the two algorithms in terms of their computational weights.

Extreme DARK is an $(R-1)$-times slower version of the algorithm of DARKer, requiring both $(R-1)$-times the computation and $(R-1)$-times the communication (with all other factors being negligible).

Even omitting the $R-1$ factor, DARK requires more communication than FinTracer by requiring signatures to be transmitted instead of just tag values. (However, signatures only need to be exchanged once, when setting up the graph, so this is only a small factor heavier than the synchronisation step of FinTracer.) Also, tags in Extreme DARK are doubly-encrypted, meaning that one may incur a size increase due to the encryption. (If one uses a second layer of ElGamal, tag size increases by 50%. Note, however, that for the purpose of Extreme DARK the second layer only needs to be a public key encryption system, not necessarily a semi-homomorphic one.)

Note 5.1. Even if communication amounts were the same, the Extreme DARK algorithm would have been practically more communications-heavy than FinTracer. This is because all communication must pass through a single bottleneck.

The greatest cost difference between the algorithms is, however, not in the communication sizes they require but in their computational costs. Extreme DARK requires each communicated tag to be encrypted and decrypted at every propagation step. That is, unfortunately, part of what provides the algorithm's security. While encryption can be performed quickly, decryption is roughly 100 times slower than refreshing, causing in total a slow-down in the algorithm by a factor of $100(R-1)$ compared with the basic FinTracer.

Note 5.2. Although encryption is the heaviest ElGamal operation, outweighing even decryption two-to-one, in practice, encryption should be considered as having almost negligible costs. The reason for this is that in our system we perform a great many "refresh" operations, for which reason we maintain a stock of encrypted zeroes that are generated offline. Much as we discount the cost of a refresh to only include its online costs, the same can be done with encrypting. The online cost of encrypting, assuming an available stock of zeroes, is, in fact, only half the cost of refreshing. Practically speaking, this makes encryption costs entirely negligible.

The same can, unfortunately, not be said about decryption, which remains a heavy process.

References

[1] Brand, M. (2022), *The Complete, Authorised and Definitive FinTracer Compendium.* (Available as: `FT_combined.pdf`)

[2] Brand, M. (2020), Improved differential privacy for FinTracer Boolean queries. (Available as: `boolean_queries.pdf`)

[3] Brand, M. (2020), Oblivious setting of source accounts in FinTracer. (Available as: `oblivious_a.pdf`)

[4] Brand, M. (2021), ElGamal key switching, AUSTRAC internal. (Available as: `keyswitch.pdf`)