A note discovery protocol.

Notes expose a 16-bit view-tag H(shared_secret || offset)[:2]

([1], [2], [3]), over which batch PIR [4] queries are made, reducing sync bandwidth of notes by up to 98%.

Details

tx = { ephemeral_pk, notes: [{view_tag, ciphertext}, ..] } view_tag = H(dh_shared_secret || offset)[:2]

offset

increments for repeated shared_secrets.

e.g. if a tx sends 3 notes to 2 addresses, the view-tags would be H(secret_1 || 0), H(secret_1 || 1), H(secret_2 || 0)

.

To ensure no privacy leakage from query patterns, each note must have the same probability of access by clients. (e.g. If an adversary spams notes to force a target to make additional queries). However, any solution that saves bandwidth must at least segment users by note rate.

In this proposal, on-chain costs for notes increase by 0.8%, and clients leak to the network that their note rate is above/below ~1/1024 (~2 notes per minute at 10 tx/s (4 notes/tx)) (to within a delay), in exchange for consuming up to 98% less bandwidth synchronizing notes.

Sync

1. Client downloads ephemeral_pk (8B/note at 4 notes/tx) and view_tag (2B) for all notes
2. Client downloads all notes larger than 256B (for now)
3. Client submits batch PIR queries for all big (850k) note-chunks
4. This lets clients handle spikes in smaller batches
5. This lets clients handle spikes in smaller batches
6. Client submits batch PIR queries for intermediate (53k) chunks to catch up
7. Client polls using batch PIR queries for small (13k) chunks to stay up to date (5.5min at 10tx/s)

Vectorized Batch PIR:

Batch	Entries (256B)	Comm. (KB)	Init (s)	Response (s)
13	13312	385	0.7s	2.3s
52	53248	385	4.4s	1.0s
832	851968	2442	330s	16.6s

There is probably room for further parameter tuning.

Limitations

Too many notes

If recieving too many notes for even the largest segment (832 of 851968), clients should wait a random delay, then begin downloading all notes starting from a random date in the past (ideally from a different server).

Serial Depth

Notes which discover other notes are not supported. A second batch PIR query (over all notes) could be added to suport 2-layer depth at a reduced rate (e.g. 1/6144 notes).

Batch	Entries (256B)	Comm. (KB)	Init (s)	Response (s)
52	319488	385	25s	1.3s