Hi Secret Community,

One more tradeoffs discussion post for today.

We have coordinated a vulnerability disclosure affecting SNIP-20 tokens. (Secret's blog.)

Our research paper is here Some mitigations against replay attacks are already present.

Even after these mitigations, the SNIP-20 transactions do not provide sender-receiver privacy. This is the "spicy prints" problem, as illustrated by the following:

[

image

1048×503 113 KB

](https://global.discourse-cdn.com/standard17/uploads/enigma1/original/2X/e/e9bd0e104d41725aa1e05b7721d493239b7ceceb.png)

[

image

1034×435 265 KB

](https://global.discourse-cdn.com/standard17/uploads/enigma1/original/2X/2/26a1267036dc19e64f21d8948c5e27eb4c830f42.png)

Right now, the storage access pattern leaked to the untrusted operating system is very fine grained, leaking the exact record being accessed. This means that SNIP-20 tokens today do not provide any receiver privacy - transfers are completely traceable from account to account.

One approach to hiding the access pattern is to insert an (optional) Oblivious RAM (ORAM) algorithm in between the trusted and untrusted codebases, which can be accessed by the contract developer through additional opcodes. This could increase the compute cost and disk I/O bandwidth (which is a bottleneck resource) for large arrays like SNIP-20 account accesses. The overhead may be too big to justify, and it may take significant development effort to get there given how Secret and Cosmos currently work. As an alternative to platform-level access pattern hiding, lightweight "obfuscation" (similar to Monero) could be implemented by SNIP-20 developers in the existing contract system. This wouldn't require platform level upgrades and may have less performance overhead, but would also leak statistical information.