CryptoNote Coin

Michael Davis

University of Miami

April 2019
Proof of Work

“scrypt” (used by Litecoin) has a uniform distribution of dependent lookups. Given a fast enough processor, it can be faster to only keep a small scratch-pad in memory and compute other dependencies on the fly.

CryptoNight ensures that new blocks depend on all previous blocks, so CPU speed trades off with memory exponentially.
Terminology

- $l = 2^{252} + 27742317777372353535851937790883648493$
- A base point $G = (x, -\frac{4}{5})$
- $H_s$ a cryptographic hash function
- $H_p$ a deterministic hash function
- Private keys are numbers $a \in [1, l - 1]$
- Public keys are points $A = aG$.
- Private user keys are pairs $(a, b)$ of private ec-keys
- A standard address is a pair $(A, B)$ derived from the private user key $(a, b)$
Table of Contents

Unlinkability

Untraceability
Alice → Bob: Step 1

Alice reads Bob’s public address \((A, B)\).
Alice → Bob: Step 2

Alice generates $r \in [1, l - 1]$ and then computes

$P = \mathcal{H}_s(rA)G + B$. 
Alice → Bob: Step 3

Alice generates the transaction. The transaction contains:

- The transaction public key $R = rG$
- The amount being transferred
- The destination key $P = H_s(rA)G + B$
Alice → Bob: Step 4

Alice sends the transaction.
Bob is listening on the stream of transactions. For each transaction, using $R$, the public key of the transaction, and $a$, one of Bob’s private ec-keys, Bob computes $P' = H_s(aR)G + B$. If $P' = P$ (the destination key), then $aR = arG = rA$. The transaction is for Bob, and only Bob (and Alice) know.
Bob computes $x = H_s(aR) + b$. Now he can use $x$ to sign an input. No one else knows $b$, so only Bob can claim the input. Bob can give away the pair $(a, B)$ (the “tracing pair”) to an auditor so the auditor can know which transactions belong to Bob. The auditor does not have enough information to compute $x$. 
Table of Contents

Unlinkability

Untraceability
One Time Ring Signatures

One time ring signatures have 4 operations

- GEN generate
- SIG sign
- VER verify
- LNK link
One Time Ring Signatures: GEN

GEN takes randomness and produces \((x, P)\), a point and number \((P = xG)\) and a point \(l = x\mathcal{H}_p(P)\).
One Time Ring Signatures: SIG

SIG takes

- $m$ a message
- $S'$ a set of public keys
- $(x, P)$ (generated by SIG)
- $I$ (generated by SIG)

and outputs

- $\sigma$ the signature
- $S$ the set $S' \cup \{P\}$

It’s producing a signature given a set of keys.
One Time Ring Signatures: VER

VER is a predicate that takes

- $m$ a message
- $S$ a set of public keys
- $\sigma$ a signature

It's verifies that the signature is an encoding of the message signed by one key in the set $S$. 
One Time Ring Signatures: LNK

LNK is a predicate that takes

- $\mathcal{I}$ all previously seen key images \{I\}
- $\sigma$ a signature

It verifies that the I encoded in the signature has not previously been seen. If it has been seen before, the signer is attempting to double spend and the transaction is invalid.