CRYPTONOTE STANDARD 004
Category: Main Track

Albert Werner
Kenji Sugihara
Montag
Ardolabar
Tereno
Antonio M. Juarez
CryptoNote
September 2012

CryptoNote Transactions

Abstract

This document is part of the CryptoNote Standards describing a peer-to-peer anonymous payment system. It defines the transfer of assets between users through transactions. Each transaction consists of inputs (i.e. references to the funds owned by the sender prior to the transaction) and outputs (i.e. records of the subsequent ownership of those funds). As a proof of ownership, the sender digitally signs the transaction with his secret keys in zero-knowledge, using one-time ring signature.

Copyright and License Notice

Table of Contents

1. Introduction

Each transaction can be considered as a transfer of asset ownership. Typically, there are two parties: the sender and the receiver. In CryptoNote, we will refer to asset as "coins" or "money".

Sender collects references to the money he is willing to redistribute into an array of inputs. Then he generates a number of keys recognizable by receiver and puts them into an array of outputs, together with the amounts. Some of the outputs will return the odd money to the sender, in case there is any.

To prove his ownership and to protect the transaction from being altered, the sender generates a signature using his secret key and attaches it to the transaction. This operation is performed for each input, so the number of signatures would be the same as the number of inputs.

2. Definitions

base transaction: a transaction that generates new coins

block: a set of data (payload) with a block header

block height: the distance between the block and the genesis block in the blockchain

double-spending: the result of successfully spending an amount of money more than once

input: a reference to an asset owned by the sender prior to the transaction

output: a new record of ownership of an asset transferred by the transaction

peer: a participant in the p2p (peer-to-peer) CryptoNote network

transaction: a single record of assets ownership transfer

transaction prefix: the part of a transaction that contains all the data except signatures

3. Transaction Structure

Each transaction consists of two parts:

1. Prefix. This part contains all the data about the previous and the new owners of the money, including how and when the funds contained in transaction can be released, and, possibly, some extra information. The whole structure must be hashed and signed by the sender.

2. Signatures. An array of signatures which prove the sender's money ownership (see [CNS002]).

Field	Type	Content
prefix	transaction prefix	Transaction data without the signatures. See section 3.1
signatures	array of signatures	Array of transaction signatures

Table 3: Transaction structure description

3.1 Transaction Prefix

The table below describes version 1 of transaction_prefix.

Field	Type	Content
version	varint	Transaction format version
unlock_time	varint	UNIX timestamp
input_num	varint	Number of inputs
inputs	array of inputs	Array of inputs. See section 3.2
output_num	varint	Number of outputs
outputs	array of outputs	Array of outputs. See section 3.3
extra_size	varint	Number of bytes in the Extra field
extra	array of bytes	Additional data associated with a transaction

Table 3.1: Transaction prefix structure description

- version: Version defines the transaction parsing rules (i.e. transaction content) and is incremented by each transaction format update. Parsing transactions with transaction_prefix of an unknown version is not safe because transaction content could be misinterpreted. Currently only transactions of version 1 are defined.

- unlock_time: This field stores the timestamp corresponding to the time the funds may be redeemed by another transaction. See section 3.4.

- inputs: This array consists of one or more inputs. See section 3.2.

- outputs: This array consists of one or more outputs. Each output is a tuple (amount, target), where targets can be of different types.

- extra: This field may store arbitrary information. Usually it is used as a part of one-time key generation process.

For varint (variable-length encoding of integers) description see section 3 of [CNS003].

3.2 Inputs

Each input contains the information about a particular sum that is used by the transaction. See [CNS002] for details on how the sender proves his ownership of the funds.

Allowed types of inputs are txin_gen and txin_to_key.

3.2.1 txin_gen

This type is used only once per block as the sole input of the very first transaction. This transaction can be created only by the peer who has found the block.

Field	Type	Content
input_type	byte	Input type. 0xff = txin_gen
height	varint	Height of the block which contains the transaction

Table 3.2.1: txin_gen structure description

See [CNS003] for details.

3.2.2 txin_to_key

This is the most frequent type of input, since it corresponds to the common case with the sole owner of the funds spending his money. Each input "spends" one of the past outputs of type txout_to_key in a way that indistinguishably hides this output among the others. To anonymously prove his ownership the sender must create a ring signature and put it into the array of signatures at the end of the transaction (see [CNS002]).

Field	Type	Content
input_type	byte	Input type. 0x2 = txin_to_key
amount	varint	Input amount
key_offset_num	varint	Number of keys used by the input
key_offsets	array of varints	Offsets corresponding to the outputs referenced by the input
key_image	key image	Image of the key of the output spent by the input, used to prevent double-spending

Table 3.2.2: txin_to_key structure description

- amount: This field stores the amount of money; this value is equal to the corresponding output's amount, which is actually being spent.

- key_offsets: The list of offsets in the global array of outputs of type txout_to_key having the same amount as the input. The first value is the ordinal number of the first referenced output among those having the same amount. Each of the following values is the offset of the next referenced output relative to the previous one. One of the outputs referenced is the actual output being spent, but only the sender known which one it is. The array of the corresponding public keys is a part of one-time ring signature verification algorithm input [CNS002].

- key_image: This field stores the image of the output's key. Each key has only one image, which is used to prevent double-spending. Only the sender can compute this value, because this process requires the knowledge of the corresponding secret key. The same value occurring in more than one input indicates that the same output is being spent more than once. Note that while it prevents double-spending, each output may be nonetheless used in any number of key_offsets as a hiding factor. The key image is a part of one-time ring signature [CNS002].

3.3 Outputs

Each output is a tuple (amount, the way how these funds can be redeemed). The sum of all outputs' amounts must not exceed the sum of all inputs' amounts.

Field	Type	Content
amount	varint	Output amount
target	output target	Output destination. Destinations can be of different types

Table 3.3: Output structure description

- amount: The amount of money being transferred to the new owner.

- target: The content of this field specifies the way the new owner can claim the money. Allowed type is txout_to_key.

3.3.1 txout_to_key

This type of output target corresponds to the most common case: a single receiver gets the right to redeem the amount specified in the output using his own secret key. The target content is therefore the corresponding public key.

Field	Type	Content
output_type	byte	Output type. 0x2 = txout_to_key
key	public key	Output public key

Table 3.3.1: txout_to_key structure description

- key: The field stores the receiver's one-time public key. Instead of using a permanent receiver's public key, the sender utilizes Diffie-Hellman protocol [DH] (using his own random data and the recipient's address) to obtain a unique key. Thus he achieves unlinkability of all keys and transactions.

3.4 unlock_time

Creating a transaction, the sender specifies the unlock_time. If this value is less than CRYPTONOTE_MAX_BLOCK_NUMBER (500000000), then it is interpreted as the block height at which the funds are unlocked. Otherwise, the value is interpreted as the UNIX timestamp.

unlock_time is used as a way to temporarily lock the funds. Precisely, each output can only be spent (or referenced by an input, even if it is not really spent) after the unlock_time elapses. Until then these funds are considered locked and non-spendable.

4. References

[CNS002] "CryptoNote Signatures", CryptoNote Standard 002, May 2012.

[CNS003] "CryptoNote Blockchain", CryptoNote Standard 003, September 2012.

[DH] Diffie, W., and M. Hellman, "New Directions in Cryptography", 1976.