Proof of multi-representation

Status: DRAFT

August 30, 2024

1 Overview

This document outlines a variant of what is commonly known as "Generalized Schnorr protocols" (see e.g. [1]). The particular variant discussed is motivated by the needs of the auditing algorithm in this code repository ("aut-ct", auditing function).

The particular variant to be analyzed imagines the following scenario:

A "representation" of a witness consisting of a tuple of m secrets x_j, with respect to a set of group elements or "bases", B_j, is defined as the group element:

$$C_i = \sum_j x_j B_{i,j}$$

Notice that this implies for N commitments $C_0 \dots C_{N-1}$, and m secrets as above, we will have a matrix of bases $B_{i,j}$ of dimension $N \times m$. Notice that **each of the commitments are using the same witness values** (hence "multirepresentation").

Correct application of the algorithm proves, in zero knowledge, that the prover knows the secret witness(es), and that each of the commitments is formed using that same witness with respect to the given bases.

The remainder of this document does the following:

- A description of each of Setup, Prove and Verify steps of the algorithm
- A proof of soundness (non-forgeability) of the algorithm
- A proof of (honest verifier) zero knowledge of the algorithm
- A brief description of a specific variant of the protocol used in auditing proofs as in this code repository, and an analysis of the security properties of this specific case.

In particular, the last section justifies that the usage of this "multirepresentation" proof in this code repository is sound (cannot be forged) and does not leak privacy information about utxos of the prover (this is the principal motivation of the document).

2 Notation

Although we are tacitly assuming the use of the elliptic curve secp256k1, the remainder should apply to any other group $\mathbb{G}$ of prime order in which the discrete log problem is hard.
We use $\mathbb{H}$ to mean specifically any cryptographically secure hash function for hashing arbitrary length byte messages. SHA-2 would be one valid possibility.
As is standard practice, we use:

- capital letters Q for group elements
- lower case letters q for scalars in the group $\mathbb{Z}_p$, where p is the order of the group
- additive notation, with + meaning elliptic curve point addition and scalar multiplication being implicit, e.g. qG.

An additional slightly nonstandard use of notation for brevity: a subscript in an equation always means that the equation holds for all valid integers which that subscript represents; more specifically:

$$j = 0 \dots m-1 \text{ and } i = 0 \dots N-1$$

with i, j defined in the previous section.

3 Algorithm definition

Setup : both parties use as inputs:

- Group elements, a set of size $N \times m$, with all relative discrete logs unknown. This is the matrix $B_{i,j}$.
- Prover shares with verifer, the **claim**: knowledge of multi-representation of a list of commitments C_i.
- An agreed string "context-label".

Prove :

Prover calculates a proof Π:

- Choose k_j at random from Z_p
- Calculates an ephemeral commitment (nonce point) $R_i = \sum_j k_j B_{i,j}$ for each commitment.
- Applies the Fiat Shamir heuristic by calculating the hash output $e = \mathbb{H}(R_i, C_i, \text{context-label})$.
- Calculates a list of j responses, as $\sigma_j = k_j + e \times x_j$.

The proof Π thus consists of N quantities R_i and m quantities σ_j.

Verify :

The verifier takes the proof Π and:

- decodes the lists R_i and σ_j from Π
- Verifies if $e = \mathbb{H}(R_i, C_i, \text{context-label})$
- Verifies if $R_i + eC_i = \sum_j \sigma_j B_{i,j}$ for every i.

Completeness can be demonstrated trivially.

4 Representation Uniqueness

Before going on to prove soundness and zero-knowledgeness, we state the following (relatively intuitively obvious, and probably covered elsewhere) theorem:

Theorem

Given a representation of a group element C with respect to a set of bases B_j, in the form of a set of scalars x_j, it is computationally infeasible to find an alternative representation set x'_j, unless the discrete log between at least two of the bases B is known.

Proof

We proceed by induction on the size of the list of bases (using consistent notation, this size is m, i.e. $j = 0 \dots m-1$. For the case $m = 2$, we can reduce knowledge of a second representation, to a discrete logarithm solving, as follows:

$$\begin{aligned} C &= aB_0 + bB_1 \\ C &= a'B_0 + b'B_1 \\ \implies B_1 &= \frac{a'-a}{b'-b}B_0 \end{aligned}$$

From here, let us assume that the theorem is true for case $m = q \geq 2$, and examine the case $m = q + 1$. Given that in this assumption, we *cannot* find an alternative representation for any q terms in the right hand side of:

$$C = x_0 B_0 + \dots x_{q-1} B_{q-1} + x_q B_q$$

, then let us consider the first q terms as a unique representation. If there is an alternative representation it therefore is required to find an x'_q such that $x_q B_q = x'_q B_q$ which is impossible because the group is of prime order, and therefore the mapping $Z_p \implies \mathbb{G}$ is one-one.

5 Computational Soundness

Comparing with the "vanilla" Schnorr identity protocol, we note a perhaps trivial difference: the same argument can be applied to only *one* of the commitments, with the others being redundant. Considering only one commitment, say $i = 0$, we can apply the standard rewinding argument to the interactive version of the protocol:
After prover provides R_0, the extractor can return different values e, e' in two separate executions, receiving in return two response *sets* σ_j, σ'_j. He can then extract a valid witness as a set x^*_j, where:

$$x_j^* = \frac{\sigma'_j - \sigma_j}{e' - e}$$

this working of course, as usual, because the k_j values in the two executions are the same.
To assert that the set $x_j^* = x_j$, we refer to the Theorem of the previous section.

6 Honest Verifier Zero Knowledge

Modelling the hash function $\mathbb{H}$ as a random oracle, and by having a Simulator $\mathbb{S}$ control the adversary's ($\mathbb{A}$) access to the random oracle, we can generate fake transcripts without knowledge of the witness, thus demonstrating the zero knowledge property, for the protocol as described. This is, again, largely identical to the base Schnorr identity protocol for a witness consisting of a single scalar value in $\mathbb{Z}_p$; the only difference is that the protocol is repeated.

The transcript of an honest execution looks like:

$$(R_i, e, \sigma_j)$$

So $\mathbb{S}$ can generate an arbitrary number of faked transcripts by choosing all σ_j values at random, then choosing the RO value e at random, then calculating $R_i = \sum_j \sigma_j B_{i,j} - eC_i$ for all commitments C_i, and then patching the RO function such that $\mathbb{H}(R_i, C_i, \text{context-label}) = e$.
Much as for the soundness argument, nothing here is actually different from the single base case, i.e the Schnorr identity protocol/Schnorr signature for a single secret key.

7 Privacy considerations in the "non-pure" case

First, let's define "non-pure".

7.1 Non-pure instances of the statement

The statement is "I have knowledge of the single representation of C_i with respect to each of the base vectors B_i, whose relative discrete logs are unknown, $\forall i$", in brief, as expanded on in detail above. In this section we focus on **what could be revealed from the C_i themselves given any additional information**, beyond the statement (because indeed, logically, the proof Π does not add to this information, by virtue of its (honest verifier + RO model) zero knowledge property.

The statement is *pure* if no other information about any of the commitments C_i is revealed. Non pure instances are non-pure in three possible senses:

1. Some portion of the witness vector is revealed, e.g. x_q $q \in 0..m-1$.
2. Some subcomponent of the commitment is revealed, e.g. $C_{i,0} = P$ such that $C_i = P + x_1 B_{i,1} + \ldots + x_{j-1} B_{i,j-1}$.

3. The $B_{i,j}$ are not all "relative-discrete-log-unknown". Specifically, we consider the case of $B_{i_1,j} = B_{i_2,j}$. Note that we do *not* consider the case where $B_{i,j_1} = B_{i,j_2}$ as that would violate the soundness assumptions described above, so that its "impurity" is irrelevant.

The implications of these "impurities" will depend on the use case of the protocol.

A variant of cases (1) and (2) is if, though the exact values P or x_j were not directly revealed, they were known to come from an enumerable set. This actually applies to our auditing use case, since we construct commitments $C = P + vG + rH$ where v is the value of the utxo and P is its public key, and they come from an enumerable set on the blockchain. This set is exponentially smaller than $\mathbb{G}$.

Given the possibility of such "impurities", we must define the additional privacy properties that we wish to claim. We define $\mathbb{P}$:

$\mathbb{P}1$ *is the claim that the value x_j^* is not revealed by the set of commitments C_i even if x_j is revealed for all indices $j \neq j^*$.*

$\mathbb{P}1$ is a direct consequence of the hardness of the DLP for the group $\mathbb{G}$.

$\mathbb{P}2$ *is the claim that: given that the values of x_j $\forall j \neq j^*$ come from enumerable sets of size X_j, such that the tuple is a set of size $\prod_j X_j$, the value of $x_j^* B_{i,j^*}$ is still not revealed for any i (that is, though the list of* ***possible*** *values of $x_j^* B_{i,j^*}$ is enumerable, it cannot be chosen).*

Small corollary: if for some index j, the group element $x_j B_{i,j}$ is revealed instead of x_j, the same result holds. Notice that this covers impurity (2) as well as (1).

$\mathbb{P}2$ is probably best understood with an illustrative example:

$$
\begin{aligned}
&C_0 = xB_{0,0} + \{Q_0\} \\
&C_1 = xB_{1,0} + \{Q_1\} \\
&... \\
&C_{N-1} = xB_{N-1,0} + \{Q_{N-1}\}
\end{aligned}
$$

where, as a shorthand, we represent the (group element) values which are chosen from an enumerable set (of size $\prod_j X_j$) using $\{Q\}$.
From this it is clear that we can enumerate the full set of *possible* values of $xB_{i,0}$ but there is no way to distinguish which is correct, which is the desired property.
This understanding of $\mathbb{P}2$ also allows us to conclude that it is not violated by "impurity" (3) above, since having the same base across different commitments is equivalent to simply knowing that group element (whether it is the same, or known, it can be subtracted/removed from the unknown components).

7.2 Application to auditing protocol

In the auditing use case we have $m = 3$ and $N = 2$, and we have a mix of impurities (1), (2) and (3):

$$C_0 = xG + vJ + rH$$
$$C_1 = xG_1 + vG_2 + rH$$

Impurity (1) applies because v is from an enumerable set (from the blockchain), (2) applies because $P = xG$ is similarly enumerable, and impurity (3) applies because we use the same blinding base H (although that could be trivially avoided). However $\mathbb{P}2$ still holds, as per the above arguments. Concretely, consider the subtraction $C_1 - C_0$:

$$xG_1 = v(G_2 - J) - C_1 + C_0 + P$$

; while the right hand is easily enumerable using all possible values of the tuple (v, P) from the blockchain, it is not possible to validate the correctness of the result for the correct specific tuple (P^*, v^*) without violating the DLP hardness assumption of the group (to put it another way, xG_1 reveals nothing about x, and cannot be correlated with P).

8 References

1. Proof systems for general statements about discrete logarithms `https://crypto.ethz.ch/publications/files/CamSta97b.pdf` Camenisch, Stadler 1997