Multiple Denominations in E-cash with Compact Transaction Data

Sébastien Canard\(^1\) and Aline Gouget\(^2\)

\(^1\) Orange Labs R&D, 42 rue des Coutures, F-14066 Caen, France
\(^2\) Gemalto, 6, rue de la Verrerie, F-92190 Meudon, France

Abstract. We present a new construction of divisible e-cash that makes use of 1) a new generation method of the binary tree of keys; 2) a new way of using bounded accumulators. The transaction data sent to the merchant has a constant number of bits while spending a monetary value \(2^\ell\). Moreover, the spending protocol does not require complex zero-knowledge proofs of knowledge such as proofs about double discrete logarithms. We then propose the first strongly anonymous scheme with standard unforgeability requirement and realistic generation parameters while improving the efficiency of the spending phase.

1 Introduction

In e-cash systems, users withdraw coins from a bank and use them to pay merchants (preferably without involving the bank during this protocol). Finally, merchants deposit coins to the bank. An e-cash system must prevent both a user from double-spending, and a merchant from depositing twice a coin. The anonymity of honest users should be protected whereas the identity of cheaters must be recovered preferably without using a trusted third party.

Divisible e-cash aims at improving the efficiency of both the withdrawal protocol and the spending of multiple denominations. The underlying idea is to efficiently withdraw a single divisible coin equivalent to \(2^L\) unitary coins. The user can spend this coin by dividing its monetary value, e.g. by sub-coins of monetary value \(2^\ell\), \(0 \leq \ell \leq L\). In this paper, we revisit the divisible e-cash approach by targeting the most demanding security model while providing a realistic parameter generation algorithm and an efficient spending protocol.

1.1 Related Work

A generic construction of divisible e-cash schemes which fulfill the classical properties of anonymity and strongly unlinkability without using a trusted third party to revoke the identity of cheaters has been proposed in [9]. The wallet is represented by a binary tree such that each internal node corresponds to an amount, i.e. \(2^{L-i}\) if the node’s distance to the root is \(i\), \(0 \leq i \leq L\). Each node in the

\* This work has been financially supported by the French Agence Nationale de la Recherche and the TES Cluster under the PACE project.
tree is related to a key such that the key of a child can be computed from the key of one of its ascendants. The main efficiency bottleneck of the practical instantiation given in [9] is that the user has to prove during the spending phase the correctness from the tree root to the target node without revealing none of the \( L - \ell \) intermediate values. As one node key is derived from its parents using a modular exponentiation, the user must prove, for each intermediate value and using proofs about double discrete logarithms, that they satisfy a certain relation. Each such proof is expensive and requires \( \Omega(k) \) group elements to be communicated in order to guarantee \( \frac{1}{2^k} \) soundness error. Moreover, the construction of the binary tree used in [9] (and previously used in [1]) is difficult to instantiate in practice [1]. Indeed, this construction necessitates to manage \( L + 2 \) groups \( G_0, G_1, \ldots, G_{L+1} \) with prime order \( p_0, p_1, \ldots, p_{L+1} \) respectively, such that for all \( 1 \leq i \leq L+1, G_{i+1} \) is a subgroup of \( \mathbb{Z}_{p_i}^* \). One possibility is to take \( p_i = 2 \times p_{i-1} + 1 \) for all \( 1 \leq i \leq L + 1 \). Using prime number theory, it is possible to show that the probability to generate such prime numbers is approximately \( 2^{-95} \) for 1024 bits prime numbers and \( L = 10 \), which is unpractical.

A very efficient variant of this scheme based on bounded accumulators has been proposed in [1]. Its main drawback is that it does not fulfills the classical security property of unforgeability. Indeed, it is possible for a malicious user to withdraw a divisible coin of monetary value \( L2^L \) whereas the legitimate value is \( 2^L \) by cheating in the construction of the binary tree of keys during the withdrawal protocol. Next, the user can spend \( L2^L \) coins without being detected and identified. The solution proposed by the authors is that the bank will use the cut-and-choose method during the withdrawal protocol by flipping a coin \( b \) and executing the withdrawal protocol correctly if \( b = 1 \) and asking the user to reveal her binary tree that is finally dropped if \( b = 0 \). If the revealed tree is correct, the user is honest and the withdrawal protocol is repeated again from the beginning. If the user is a cheater, a fine of value \( 2L2^L \) is deducted from the user’s account. This drawback may be considered as unacceptable from the bank point of view even if the bank should not loose money “on average”.

1.2 Our Contribution

We revisit the divisible e-cash approach by targeting both the most demanding security model and the effective possibility to instantiate an e-cash system from a theoretical method. We introduce a new construction based on algebraic objects to generate the binary tree without any previously mentioned problems (impracticability of the key generation [9] and unusual security model [1]). We introduce a new technique to prove the validity of the spending. We show that it is possible to prove that one node key is derived from its father, which is impossible in the proposal of [1]. This enables us to prove that one node key is derived from only its father and we do it only once instead of \( (L - \ell) \) times in the scheme proposed in [9] for spending \( 2^\ell \) coins from a divisible coin of \( 2^L \) coins. Next, we prove the remainder of the paths from the spent node to the leaves using a variant of the accumulator technique from [1]. In our construction, the spender only sends to the merchant a constant number of bits to spend \( 2^\ell \) coins.