Sipster: Settling IOU of Smart Meters Privately and Quickly

A. How to run the code?

It only requires three steps to run the Sipster. First, we need to install two tool kits, i.e., PBL and OpenSSL. We use ECDSA from OpenSSL. Its elliptic curve is over a prime field of n = 256 bits. We use the Pairing based Cryptography Library (PBL) with a "Type A" elliptic curve generated for 256-bit group order and 512-bit base field. Second, we need to download the source file from the code repository and compile the source files in the folder "Codes" using the GCC compiler. It gives a runnable output file "Sipster.run". Third, run the "./Sipster.run" to test the algorithm.

PBL: http://crypto.stanford.edu/pbc
OpenSSL:https://www.openssl.org/

B. How the code matches with Sipster's algorithms?

The code contains seven functions that match with the critical procedures described in section 4.3. We use the blue text to mark the functions used in the code and explain how they fit with specific procedures as follows.

1. Setup Phase:
In this phase, the smart meters (SM), utility company (UC), and residential users (RU) will initialize their critical parameters. We complete these initializations in the main function of the Sipster.

2. Bill Issuing Phase:
SM updates its internal state τ and records the real-time consumption of RU and calculates the fine-grained expense. For each unit of expense, SM runs the stateful algorithm TokenGen (Algorithm 1). We use the SM_TokenGen function to describe these steps.

3. Bill Settlement Phase:
Once RU receives a token tk in the Bill Issuing Phase, RU can settle the bill for this particular token and receive receipts from UC (Algorithm 2). Receipts

are then verified by RU to ensure their correctness. We use the UC_ReceiptGen function and RU_Verify function to achieve these procedures.

4. Bill Verification Phase:

In the billing period t, SM generates K tokens. Thus, RU collects K receipts from UC. To verify all bills are settled, SM first needs to prepare bill information (Algorithm 3) for RU. We use SM_BillGen function to generates the bill. Then, RU combines all his receipts and submits them to UC as proof of bill settlements (Algorithm 4). We use RU_CombineReceipt function to combine the receipts. And, UC_Bill_Verify function finally verifies the bill settlements.