Smart Meter (SM) 1
1. Selects pairing-friendly curve: $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ of prime order p. 2. Selects a generator $P \in \mathbb{G}_2$. 3. Selects the master secret key $msk = s \in \mathbb{Z}_p^*$ at random. 4. Computes the master public key $P_{pub} = s \cdot P \in \mathbb{G}_2$. 5. Defines hash functions: • $H_1 : \{0,1\}^* \to \mathbb{G}_1$ (Hashes ID to a point) • $H_2 : \{0,1\}^* \to \mathbb{Z}_p^*$ (Hashes message to a scalar) 6. Defines standard ECC params for ECDH: Generator G of prime order q. 7. Publishes global params: $params = (e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, P, P_{pub}, H_1, H_2, G, q)$.
Smart Meter (SM) 1
Authenticates to PKG, provides its unique identity ID_{SM}. **Send:** ID_{SM} (via secure, out-of-band channel) 1

Smart Meter (SM) 1
Receives and securely stores its private key sk_{SM}. 1
Smart Meter (SM) 1

Smart Meter (SM) 1
Step 1: Initiation: 1. Selects fresh ephemeral secret $d_{SM} \in \mathbb{Z}_q^*$. 2. Computes ephemeral public key $Q_{SM} = d_{SM} \cdot G$. 3. Generates fresh timestamp T_{SM}. 4. Creates message $M_1 = (Q_{SM} \parallel T_{SM} \parallel ID_{DCU})$. 5. Computes hash $h_1 = H_2(M_1) \in \mathbb{Z}_p^*$. 6. Computes IBS signature $sig_{SM} = h_1 \cdot$

Smart Meter (SM) 1
Step 1: Verification & Response:1. Receives M1: $(ID_{SM}, Q_{SM}, T_{SM}, sig_{SM})$.2. Checks if T_{SM} is valid (e.g., $\lvert T_{now}$

Smart Meter (SM) 1
Step 2: Verification & Key Computation: 1. Receives M2: $(ID_{DCU}, Q_{DCU}, sig_{DCU}, MAC_{DCU})$. 2. Computes $Q_{ID\_DCU} = H_1(ID_{DCU}) \in \mathbb{G}_1$. 3. Recomputes $M_2 = (Q_{DCU} \parallel Q_{SM} \parallel T_{SM} \parallel ID_{SM})$. 4. Recomputes $h_2 = H_2(M_2)$. 5. **Verifies DCU Signature:** Checks $e(sig_{DCU}, P) \stackrel{?}{=} e(h_2 \cdot Q_{ID\_DCU}, P_{pub})$. (If check fails, aborts). 6. Computes premaster

Smart Meter (SM) 1
Step 3: Final Verification:
Session Established. (Securely erase d_{SM}). 1