# PayPlace: Secure and Flexible Operator-Mediated Payments in Blockchain Marketplaces at Scale

Madhumitha Harishankar, Dimitrios-Georgios Akestoridis, Sriram V. Iyer, Aron Laszka, Carlee Joe-Wong, Patrick Tague
Carnegie Mellon University, Flipkart, University of Houston

## Problem

Crypto payment protocols not optimized for marketplace-style frequent transactions between consumers and merchants.

- **Payment Channels** optimized only for pairwise transaction between two entities
- **Lightning Network** not optimized for frequent transactions in one direction
- **Payment Hubs** impose high liquidity requirements on the hub intermediary
- **Plasma-style sidechains** require users to be *online* (often to participate in *exit games)*

## Our Goal

Develop a **cryptocurrency payment protocol** optimized for marketplaces with the following properties:

- Merchants **withdraw confirmed funds** without exit games
- Merchants and consumers at **no risk of having their confirmed funds stolen** even if arbitrarily offline
- Consumer **can use their entire funds** at any time for payment to a single merchant **without relying on intermediaries' liquidity**
- Number of root-chain transactions **sub-linear** in the number of payments and recipient merchants, atleast in the optimistic case.
- Merchants **do not incur counter-merchant risk** in the process of receiving payments from consumers.
- Resistant to **Data Availability attacks.**

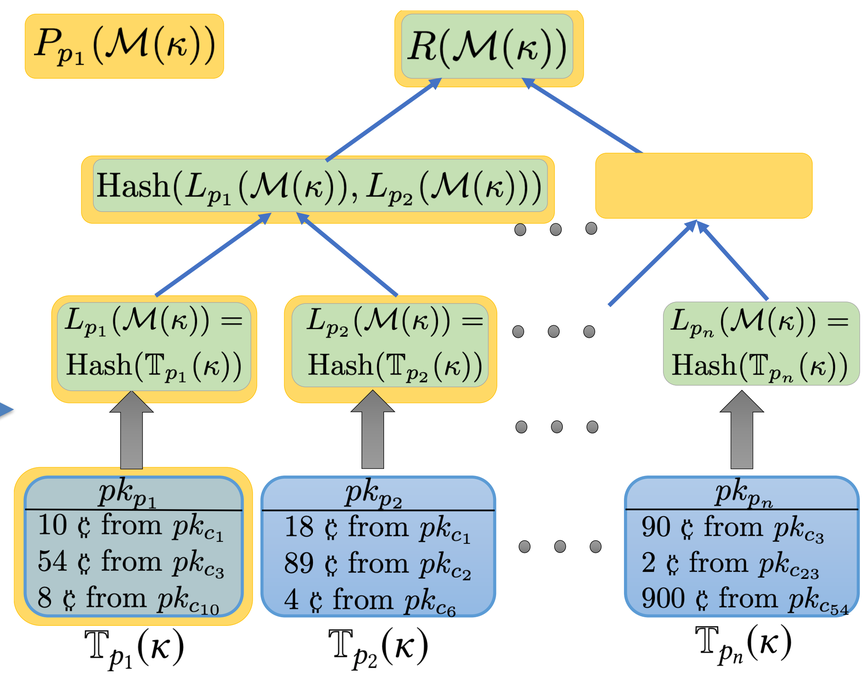
## Design

**STEP 1**

- Consumers deposit funds into the PayPlace smart-contract
- Consumers make instant off-chain payments to the *marketplace operator* specified in the contract for their marketplace orders

**STEP 2**

- Periodically, the *marketplace operator* computes a tree that specifies
  - For each merchant p
    - For each consumer c who placed an order for p since p last withdrew funds
      - Total amount owed to p from c since then
- Operator broadcasts this tree to merchants




**STEP 3**

- Merchants protect themselves against double-spend attacks from the operator by performing validation checks on the tree.
- If validation checks pass, they sign the root and send it to the operator

- **Custodial – risk capped to one aggregation period**
- **No operator deposit required**
- **No ZK Proof or Exit Games**

**STEP 4**

- Operator verifies received **BLS** signatures, aggregates them and submits the root and agg. sig. to the contract.
- Root accepted if the contract is able to verify the aggregate signature against the aggregate public key on file.
- Otherwise, contract identify merchants whose signatures are missing from the submitted commitment despite no record of their individual public keys and assures safety of their already confirmed funds

## Evaluation – Zero Knowledge Rollups as Baseline

### Comparing Off-Chain Computational Load and Runtime

| | Runtime | | # Pairing & Exp. | |
|---|---|---|---|---|
| | PayPlace | ZKR | PayPlace | ZK Rollup |
| Op. | $O(p_r)$ | $O(n)$ | $2p_r$ | $\frac{n}{z_{\max}}(4g' + w' - l')$ |
| Mer. | $O(c_u \cdot p_r)$ | $O(1)$ | $2c_u$ | 0 |

Gas cost for ZK Rollup increases with $n$ while worst-case PayPlace ( $p_m$ = 0) is orders of magnitude cheaper when $p_m$ is low w.r.t $n$.

### Comparing Estimated On-Chain Gas Costs