

# Not Yet Another Digital ID: Privacy-Preserving Humanitarian Aid Distribution

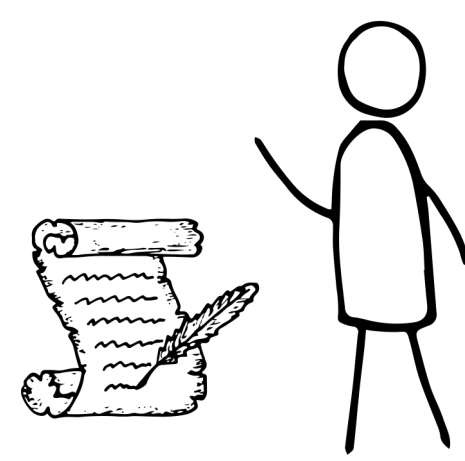
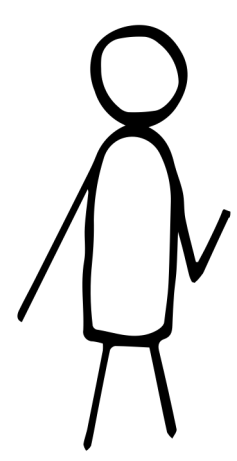
## Aid Distribution & Challenges

Humanitarian organizations distribute physical goods to people in need. Traditional aid-distribution systems are paper-based, which do not scale to large populations. **Considering the special working context, humanitarian organizations must digitalize aid-distribution systems without harm.**

Efficiency matters! People gathering together for a long time is dangerous, terrorists are around! (D2)

### Registration

1. Hi, I am @#\$. I come to register my household.
2. Okay, I confirmed, you are in the household with %^&.
3. I registered your household by writing your name, your monthly entitlement on the list.



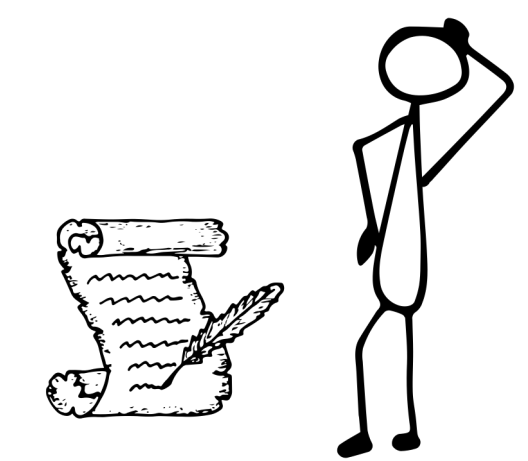
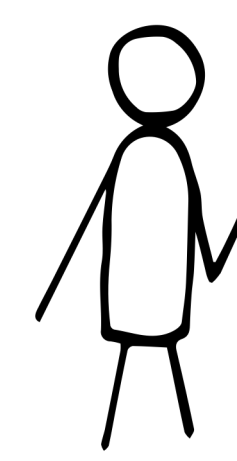
Minimize the info revealed to any involved party (P1, P2)

What if people "double-dipping" for getting more goods than they are entitled? (S1)

Only enable legitimate recipients to request aid (S2)

### Distribution

4. Hey, I come to pick up my aid for this month.
5. Hmm... let me check. (5 mins later) Ah, here you are: household of @#\$, two bags of rice.
6. Here are your bags! Sign your name here please.



Accountability? Anyone can forge signatures! (S3)  
Auditing privacy? Can we just give the list to any auditor? (P3)

Apart from **Security&Privacy** concerns, there are other Functional and Deployment requirements as well:  
F1: Distribute per household.  
F2: Enable modification.  
F3: Periodic distribution.  
D1: No stable connectivity.  
D3: Robust distribution.  
D4: Usability.

## A Token-based Design

### Main Ideas

**Decentralization** by putting personal info into tokens, people can opt out by destroying e.g., the smartcard

**Modification** via blocklist-based revocation and re-issuance to adapt to changes

**Double-dipping** prevention by checking tags, the tag is random but unique per household in one distribution period

**Trusted Execution Environment\*** (TEE) to ensure correctness of the computation

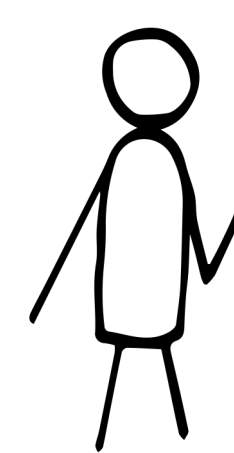
0) Manufacturer builds a private key  $sk$  into all cards

- 1) RS uses help from village elders to decide entitlement  $ent$
- 2) RS generates a revocation value  $v_H$



Registration Station (RS)

3) RS sends  $ent, v_H$  to card



5) Card checks not on BL, computes tag  $t = \text{PRF}(k_H, e)$ , Pederson commitment on entitlement  $com\_ent$ , signature  $\sigma = \text{Sign}(sk, t || e || com\_ent || H(BL))$

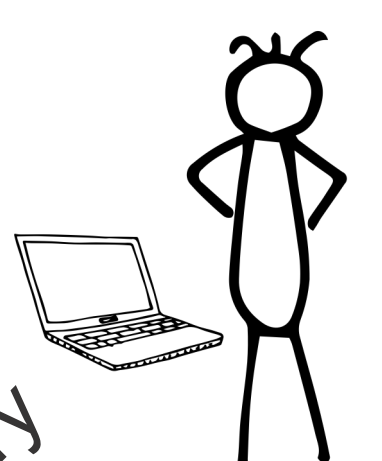
6) Card sends to DS  $ent, t$ , a proof containing  $\sigma$

4) DS sends period  $e$ , blocklist BL to card



Distribution Station (DS)

8) DS sends a summary with proof to the auditor

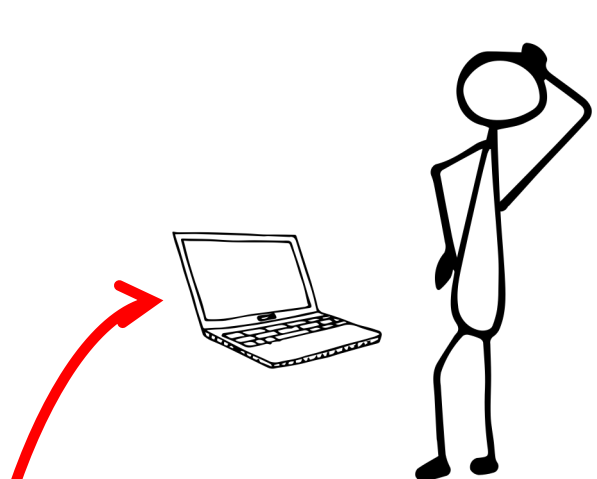


7) DS checks  $t$  for double-dipping prevention, validates the proof, and keeps records for auditing

**\*What if there is no TEE?** In some areas where aid distributions happen, smartphones are available at least per household. It is practical to use phones as tokens. However, phones are not tamper-resistant, and hence, cannot be treated as TEE. To make sure the correctness of computation happen in the phone without compromising user privacy, we incorporate **Attribute-Based Credential (ABC)** scheme into the solution. The phone runs ABC with RS to get a credential at registration, then shows the credential with zero-knowledge proofs at distribution. Talk with us for more details!

## Towards Accountability & Deployment

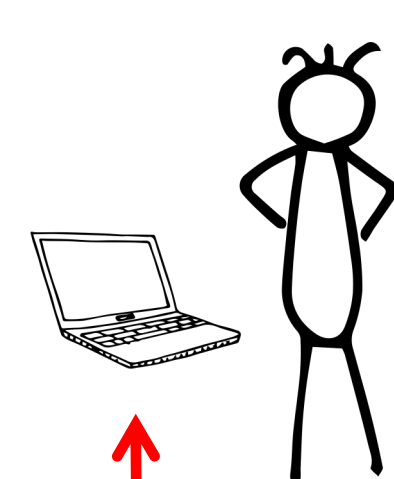
### Privacy-preserving Auditing for Accountability



- 8) Audit distributions in period  $e'$
- 10) DS sends  $sum\_ent, sum\_r, H(BL)$ , and all entries  $(\sigma, t, com)^i$

9) DS gathers all entries of records  $(\sigma, t, e', com, ent, r)^i$ , multiply all commitments to get the sum of entitlement  $sum\_ent$  and of random numbers  $sum\_r$

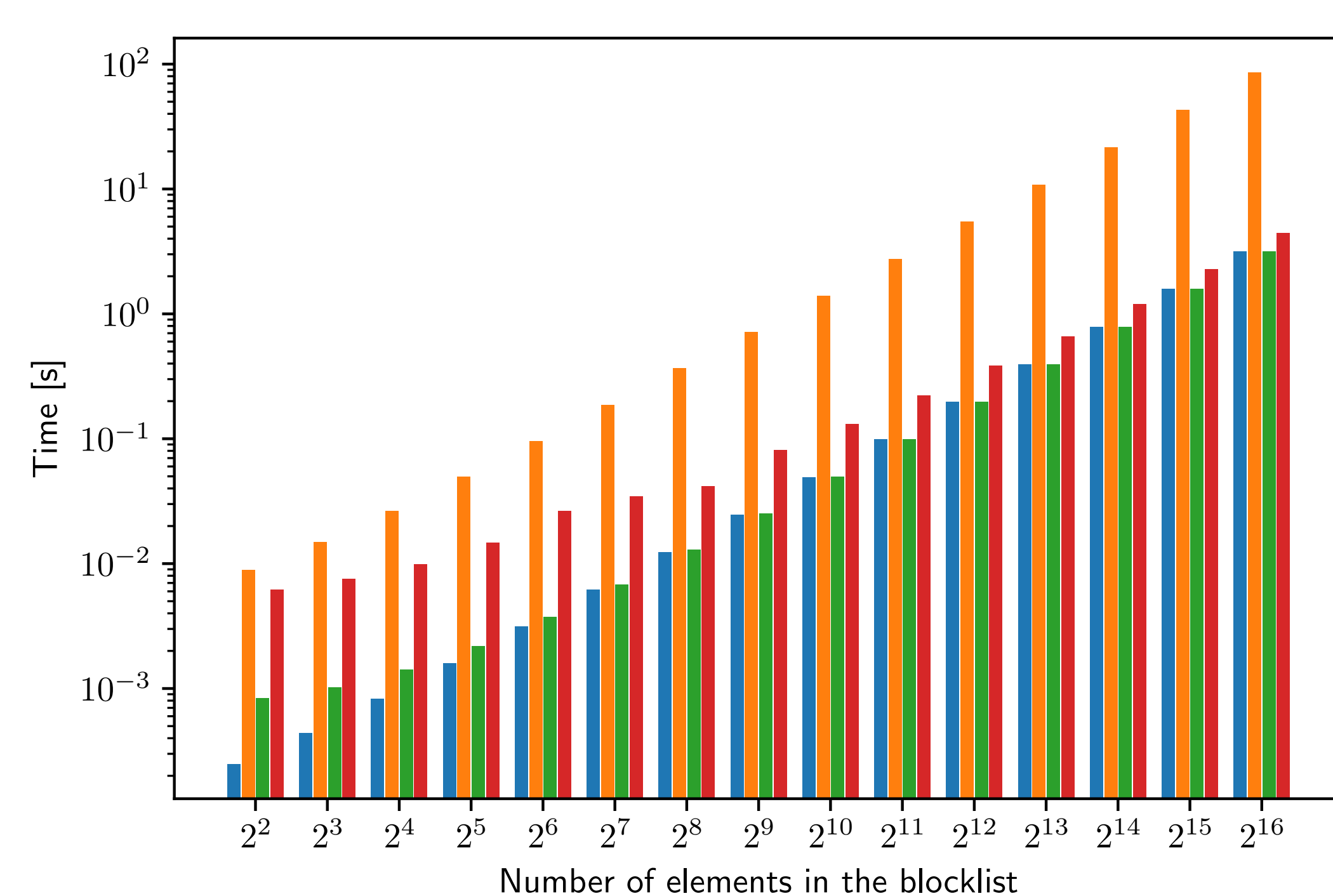
A malicious DS cannot forge signatures to fake proofs with larger sum. Accountability!



- 11) Auditor verifies all records of signatures by running  $\text{Verify}(pk, \sigma, t, || e' || com || H(BL))$ ; If verified, auditor multiplies all commitments and check the sum against  $sum\_ent, sum\_r$

A malicious auditor cannot learn the per-household  $ent$ , only the sum. Privacy!

### Performance Evaluation



Our analysis shows the card solution can finish in seconds even with 1024 entries on BL. The figure shows the phone solution also finishes fast enough.