Privacy-Preserving Reputation-Based Lending Systems

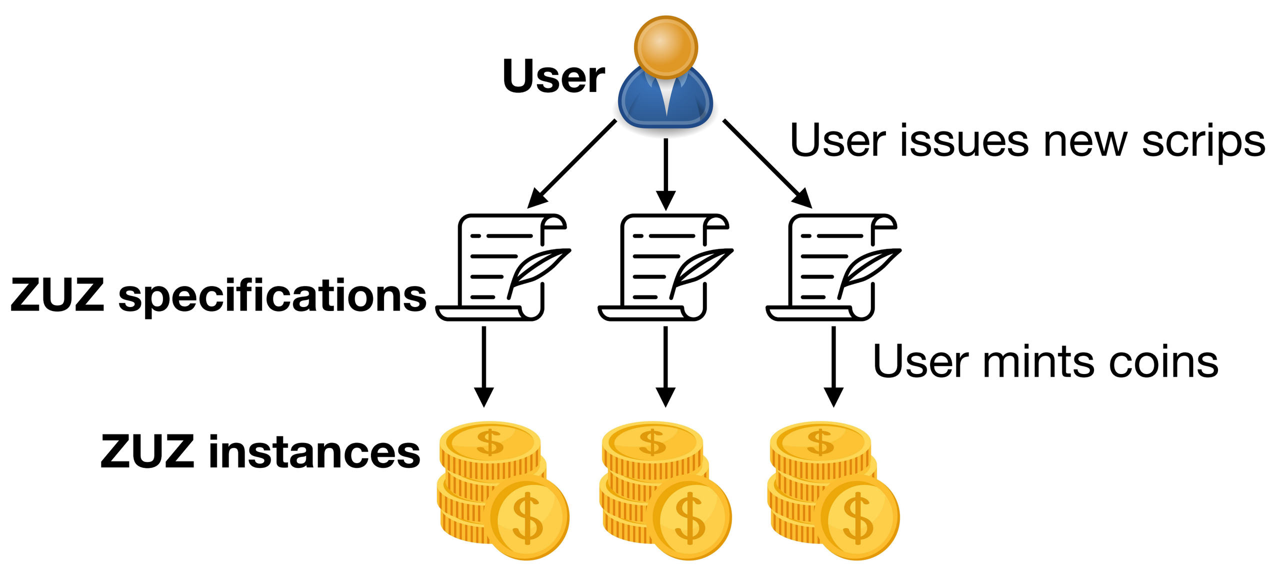
Sarayu Namineni, advised by Prof. Seth Goldstein

Carnegie Mellon University, Computer Science Department

ABSTRACT

Concerned with the privacy of their digital transactions, users are averse to leaving the very digital trace necessitated by reputation-based lending systems. Our design converts between account and UTXO wallet representations to hide the transfer's origin, value, and destination while the underlying scrip still gains reputation. Our approach results in fast and cheap online computation, while total transaction times scale linearly in the number of recipients.

BACKGROUND



In a *reputation-based lending system*, users can issue their own scrips, or *ZUZ specifications*, to be redeemed in exchange for goods and services in the future. In turn, these specifications gain value when *instances* of them are traded by individuals or at businesses outside of the ones that issued them. When such transactions are made transparent and universally accessible, such as by being recorded on a *distributed public ledger*, we can base a scrip's reputation off of its acceptability.

Under such a system, a user can conduct an anonymous transaction by minting ZUZ instances from a new specification issued under a new pseudonym. Since neither the specification nor the pseudonym have any history associated with them, the transaction is completely anonymous; however, for the very same reason, other users on the network have little to no incentive to accept these funds in a transaction. Without a way for users to spend their existing funds privately, such a system is rendered unusable.

INTERFACE

In our smart contract implementation, users can create ZUZ specifications, and mint, *pour*, and transfer ZUZ instances. A pour operation allows a user to convert a public ZUZ instance into a private ZUZ instance, and vice versa.

Public instances are represented by numerical balances, like accounts, whereas private instances are represented as lists of commitments, like unspent transaction outputs.

Note that the ZUZ specification is made public in all parts of the interface so that the underlying scrip can still gain reputation through private transfers.

Pour $(spec, b_{pub}^{old}, [cm_1^{old} \dots cm_w^{old}], b_{pub}^{new}, cm^{new}, \pi_{POUR})$: For the ZUZ specification $spec$, the sender converts between their

- old public balance b_{pub}^{old} and private balances $[cm_1^{old} \dots cm_w^{old}]$, and
- new public balance b_{pub}^{new} and private balance cm^{new}

by providing a zero-knowledge proof π_{POUR} which shows

- the sender can unlock all the provided commitments, and
- balance is preserved

Transfer $(spec, [cm_1^{recv} \dots cm_r^{recv}], [cm_1^{old} \dots cm_w^{old}], cm^{new}, [Enc_{pk_1}(\{b_1, \omega_1\}) \dots Enc_{pk_r}(\{b_n, \omega_r\})], \pi_{TRANSFER})$: For the ZUZ specification $spec$, the sender

- adds private balance cm_i^{recv} to recipient pk_i along with the parameters to unlock the commitment $Enc_{pk_i}(\{b_i, \omega_i\})$
- replaces sender's private balances $[cm_1^{old} \dots cm_w^{old}]$ with new private balance cm^{new}

by providing a zero-knowledge proof $\pi_{TRANSFER}$ which shows

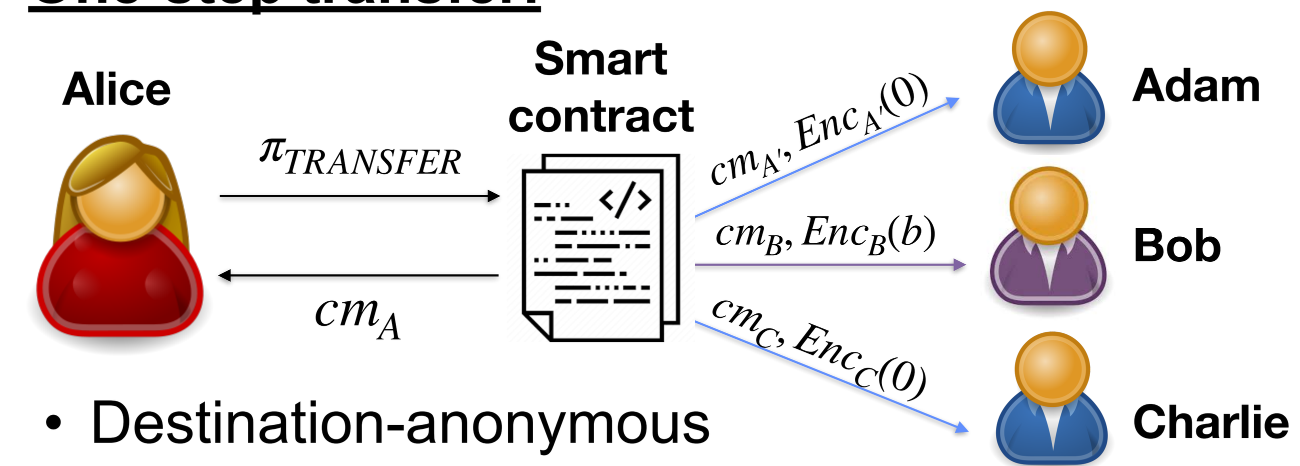
- the sender has sufficient funds, and
- balance is preserved

In both cases, the smart contract must validate that the inputs to the ZKP match the ledger state to prevent double-spending attacks.

One remaining challenge is mitigating *wash attacks* on the reputation of ZUZ specifications under private transfers.

TRANSFER PROTOCOLS

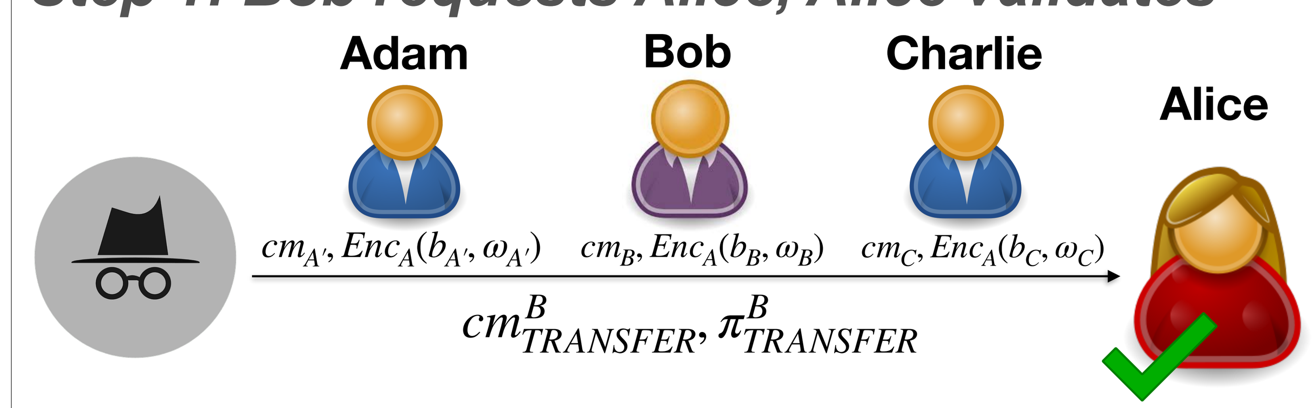
One-step transfer:



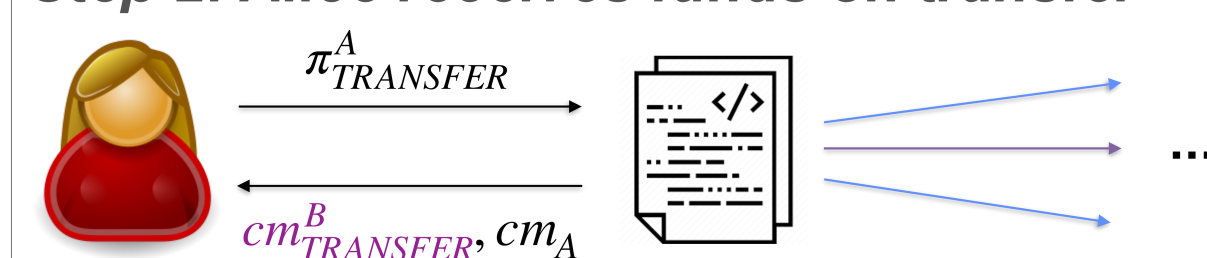
- Destination-anonymous
- Smart contract validation
- **Issue:** Alice pays for Bob's privacy

Two-step transfer:

Step 1: Bob requests Alice, Alice validates



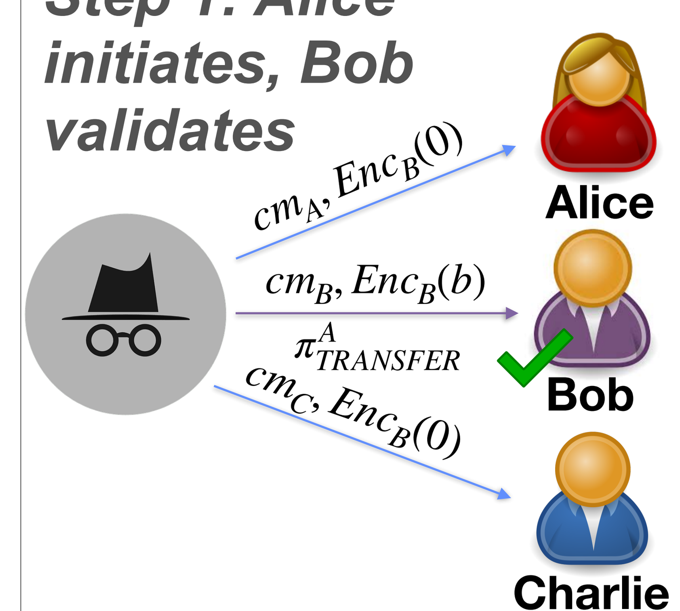
Step 2: Alice receives funds on transfer



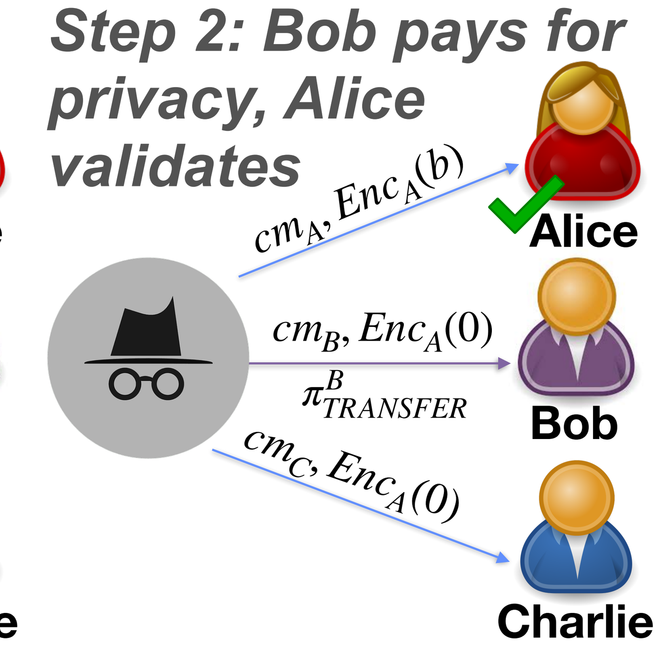
- Destination-anonymous
- Smart contract validation of Alice and client-side validation of Bob
- Bob pays for his own privacy

Three-step transfer:

Step 1: Alice initiates, Bob validates



Step 2: Bob pays for privacy, Alice validates



Step 3: Alice receives funds on transfer

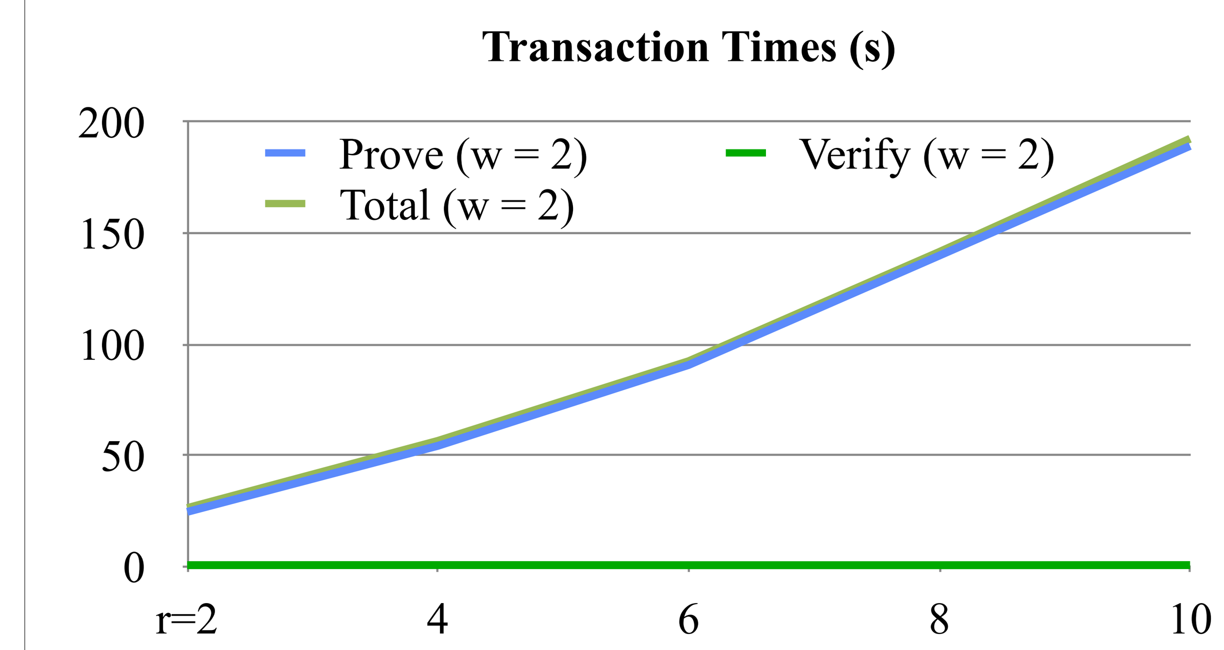
- Destination- and origin-anonymous
- Client-side validation of Alice and Bob
- Bob pays for his own privacy

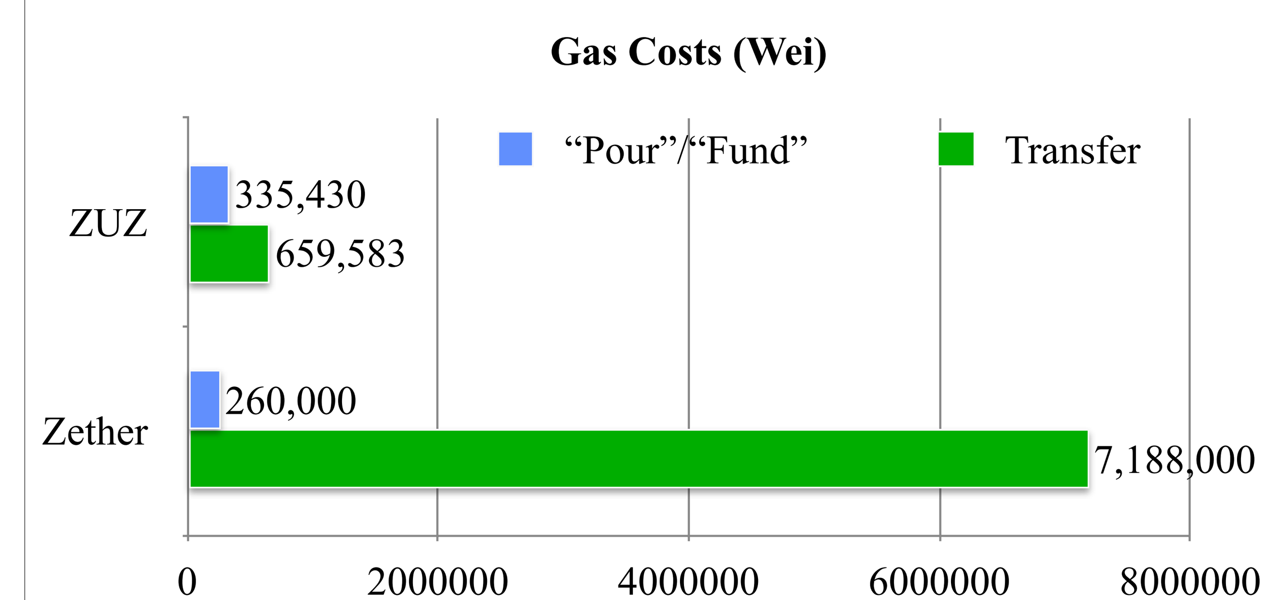
RESULTS

Our experimental results confirm that offline transaction times scale linearly in the size of the anonymity set whereas online computation is constant and low cost.

Time complexity of ZKP circuits for interface

ZKP circuit	Parameters	Time Complexity
Pour	• *w* balances	*O(w)*
Transfer (one-step)	• *w* sender balances • *r* recipient balances	*O(r+w)*





CONCLUSIONS

Moving computation offline results in practical smart contract transaction speeds and gas costs for destination- and value-anonymous transfers. By introducing client-side validation, our protocols allow senders to conceal their identity and recipients to pay for their privacy.

Future work aims to reduce the time complexity of client-side proof generation, through Merkle tree wallet representations or pre-compilation of ZKPs and analyze the system against various attack vectors (i.e. malicious clients, statistical inference, etc.).