UnTraceable Transactions (UTT) with accountable privacy and technological experimentation with the Bank of Israel

Ittai Abraham, VMware Research

Wannabe cryptographer

Some of my best friends are cryptographers

Joint work with

Alin Tomescu (Aptos, work done at VMware)

Adithya Bhat (VMware and Purdue)

Benny Applebaum (VMware and TAU)

Guy Gueta (VMware)

Benny Pinkas (Aptos, work done at VMware and BIU)

Avishay Yanai (VMware)

UTT: Decentralized Ecash with Accountable Privacy

https://eprint.iacr.org/2022/452.pdf

Why Privacy Matters

Fundamental challenge for humanity in the 21st century

Two dystopian extremes:

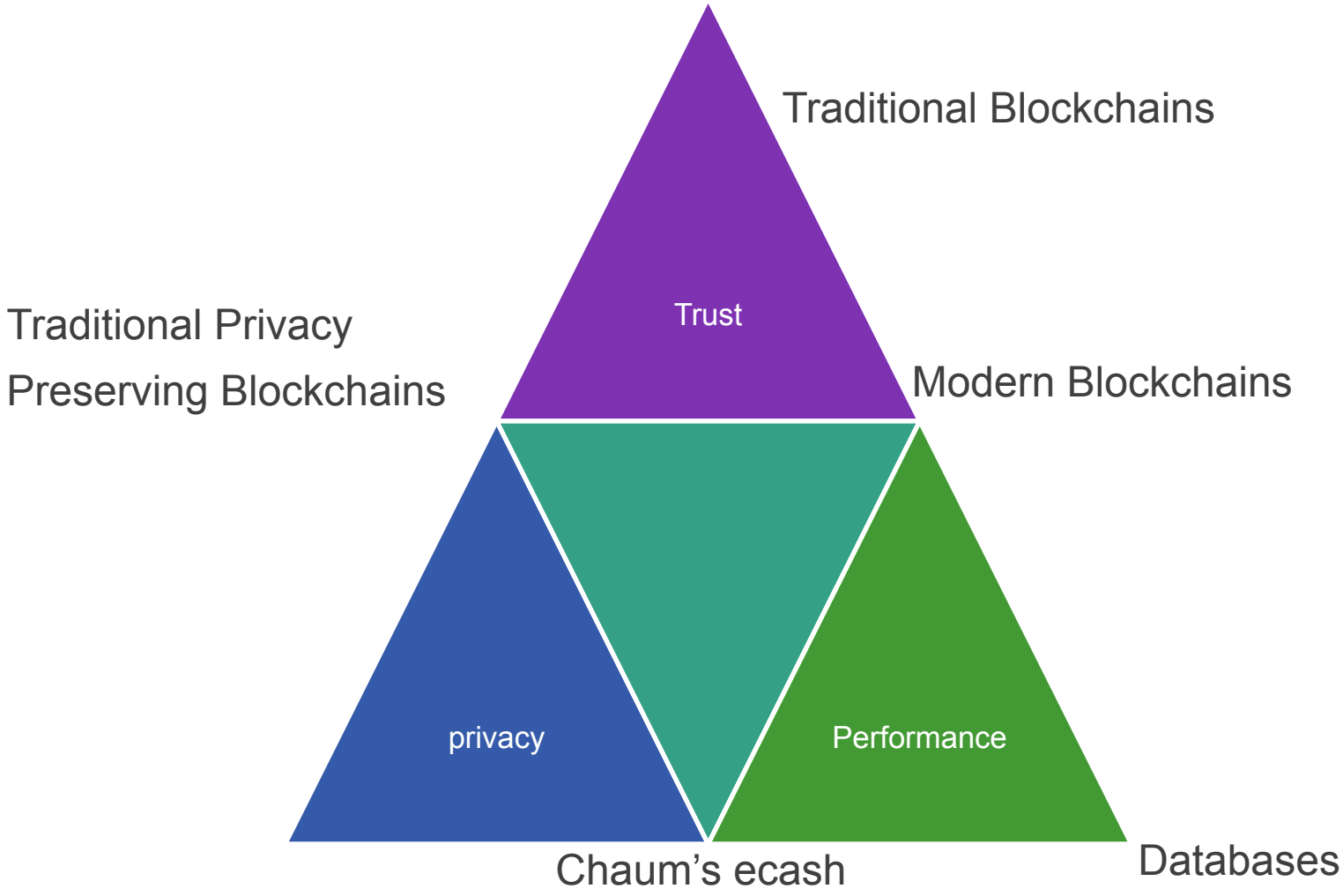
- No privacy: “bitcoin is like twitter for your bank account”
- Privacy maximalism: no accountability, no KYC, no AML, even for the DPRK

More study of sweet spots

Balance privacy with other core values like rule of law

Accountable privacy as an example

My mental model



Explorations in accountable privacy with trust and performance

Path not taken:

- Augment zcash with accountable privacy, and make it scale
  - Challenge 1: prover space after state updates
  - Challenge 2: horizontal scaling and sharding

Path taken:

- Augment chaum's ecash with accountable privacy, and make it scale
- Obtaining trust via BFT and lightweight MPC (threshold signatures)

Scalability and Privacy

What size is not size?

Backend Goals

- Want a SNARK for circuit-SAT or or other IR for which:
  1. P runs in linear time in the size of the circuit, or as close as possible.
  2. Proof size is as small as possible.
  3. V is as fast as possible.
- Also ideal:
  4. Transparent (no “toxic waste” produced in a trusted setup).
  5. Plausibly post-quantum secure.

Scalability and Privacy

What size is not size? prover space

Say you receive a coin, go to sleep for a year and then want to pay

In a system doing 1000 tx/sec – about 30B tx a year

Need to generate a SNARK of your coin in a **recent** Merkle tree

Option 1: Stream those 30B tx to build an updated Merkle tree

Option 2: Query a service just for your Merkle path

Scalability and Privacy

What size is not size? Parallel computing

Fundamentally the only way to scale is to avoid sequential bottlenecks

Can you Horizontally scale zcash? Run it on multiple shards in parallel?

Naively: requires a different Merkle tree per shard

Motivation to issue a Bank of Israel digital currency

Bank of Israel's Steering Committee, 2021

1. Offer a competative basline
2. Enable innovative use cases, smart contracts, programmable mony, DeFi
3. Redunancy, propoer function during emergency, national security
4. Efficient and inexpensive cross broarder payments
5. Maintaining the public's right for digital payment with certain level of privacy
6. Support policy to reduce the use of cash and "shadow economy"

Digital Cash Infrastructure

Digital cash should be better than physical cash

Physical cash:

- **Trust**: has physical protections for double spend prevention and counterfeit resistance
- **Accountable Privacy**: small transactions are anonymous, large transactions get recorded
- **Performance**: great for physical encounters, bad for online

Research question:

Can we build a digital cash infrastructure that has similar/better properties?

Digital Cash Infrastructure

better trust, better trade-off between privacy and accountability, and better performance

Digital Trust:

- Byzantine Fault Tolerance State Machine Replication (SBFT) for safety and liveness
- Threshold cryptography (MPC) for security and privacy

Digital Accountable Privacy:

- Entities have a digital ***anonymity budget*** that can be periodically replenished
- Transactions within this budget remain completely anonymous
- Transactions beyond the anonymity budget require real-time auditor approval
- Based on Ecash and Zero-Knowledge technology

Performance:

- 10^3 TPS via vertical scaling: Using highly efficient zero-knowledge proofs, commutativity, and modern multi-core servers
- 10^4 TPS via horizontal scaling: Multiple systems working in independently
- 10^5 TPS via hierarchical scaling: Adding privacy-preserving payment channels
- Better performance is also about enabling programmability, which we will support

Performance of UTT-based digital cash infrastructure

New, **fast** zero-knowledge proofs (ZKPs) for private payments

67 milliseconds to create a TXN

- Seconds in Zcash
- 4 kb of state

400 transactions per second (TPS)

- On a 4 server BFT system
- Using vertical scaling

1,000 TPS using a BCB system

11,000 transactions per second (TPS)

- Using horizontal scalability (i.e., sharding)

Cryptography:

- Threshold re-randomizable signatures over re-randomizable commitments
- Σ-protocols
- Zero-knowledge range proofs
- Pseudo-random functions
- Identity-based encryption
- libutt: 7,500 lines of C++
- Extensive security proof

Distributed Computing:

- Byzantine Agreement w pre-execution
- Byzantine Consistent Broadcast
- Threshold PS signatures

Bank of Israel

Digital shekel project – full document (20/6/2022)

- For the purpose of the second stage in the experiment, **a VMware blockchain infrastructure was set up in an AWS cloud that supports zero knowledge proof technologies for limited privacy**
- The system was built using a **two-tier model**, with a Byzantine Agreement system based on **VMware Blockchain and a VMware Decentralized Cash Infrastructure payment engine**
- … it is therefore necessary to work with other mechanisms involving **zero knowledge proof technologies**
- An examination of an innovative development of this technology showed that it is **possible to implement a policy whereby a periodic budget of "private" digital shekels can be allocated to each customer on the digital shekel network**, which the customer can use to pay without any documentation of the payment being kept

The 5 actors

For a two-tier (intermediated) retail CBDC with accountable privacy

1. **The Central Bank:**
   - Can mint tokens by signing them
   - Can burn tokens by adding them to its nullifier list
   - Implemented as a BFT SMR ledger and threshold signatures
2. **Users:**
   - Own tokens, have a public identifier
   - Secret key for spending tokens and receiving messages
3. **Intermediaries:**
   - Store the tokens for the users
   - Intermediates communicate between users and the central bank
4. **Registration Authority:**
   - Can create identity tokens
   - Can revoke/recreate/refresh identity tokens
   - Implemented as a BFT SMR ledger and threshold signatures
5. **Auditor:**
   - Provides budget tokens to users via intermediaries
   - Audits transactions above budget

Basic structure for Chaum’s ecash

Basic protocol:

- The central bank signs a token for Alice
- When Alice pays Bob, Alice's token is nullified (to prevent double spending), and a new token is signed for Bob by the central ban

Desired properties:

- Alice and Bob cannot generate tokens out of thin air
- The central bank cannot identify that a token belongs to Alice (even though the bank previously signed this token for her)
- The transaction is private as long as Alice does not use all of her privacy budget

Digital Token in UTT

A digital token has four parts *T=(sn, id, type, val)*

- *sn*: a unique serial number of the token
- *id*: the name of the owner of the token
- *type*: the type (regular or budget for now)
- *val*: monitory value that the token contains

Two events for a digital token:

- Mint event: a token [T] is **minted** by being **signed** by the Central Bank
- Burn event: a token [T] is **nullified** by being **added** to Central Bank **nullifier list**

Re-randomization

Users receive signed tokens and their signed identity token

- The Central Bank, which signed the token, knows which token it signed
- The Registration Authority, which signed the identity token, knows to which user it belongs to

For privacy, UTT uses ***re-randomizable signatures***:

- Users can re-randomize the original signatures they received and generate a new signature for the same token
- This new signature can be verified, but is un-linkable to the original signature

3-in and 3-out: an example UTT transaction

Alice has two tokens of $50 and wants to pay Bob $70. Alice has a privacy budget of $500

Three incoming tokens:

- T_1 – first token of Alice (with value of $50)
- T_2 – second token of Alice (with value of $50)
- BT – privacy budget token of Alice (with value of $500)

Three outgoing tokens :

- T_b – outgoing token for Bob (with value of $70)
- T_a – outgoing token for Alice (with value of $30)
- BT’ - new privacy budget token of Alice (with value of $430)

The proof verifies that $T_1+T_2=T_a+T_b$, and that $BT'=BT-T_b$

Next steps and ongoing work

Innovation thrust:

- EVM (Ethereum) integration – Call UTT from regular Solidity smart contracts!
- DeFi with accountable privacy: uniswap, oracles, lending/borrowing in a regulated CBDC
- DvP, digital checks, automatic tax payment with accountable privacy
- Improve performance, optimized budgets, better parallelization
- Payment channels with accountable privacy
- Strengthen security proof and properties

Commercial thrust:

- Central Banks: digital cash and accountable privacy for Central Bank Digital Currency (CBDC)
- Banks, Financial Institutions: digital cash and accountable privacy for Stablecoins
- Enterprise (esp. Healthcare): using ZKP and accountable privacy for non-payment use cases

Thank you!