Enabling Micro Transactions Between IoT Devices

Manoranjith A P
Senior Software Engineer, Bosch
“An electric car wanting to charge at a charging station”
Humans

- Arrive at the charging station
- Make an informal agreement
- Charge the car
- Make payment (pre/post)
- Leave the station

Machines

- How to make agreement?
- How to handle disputes?
- How to design for IoT scale?
- How to deal IoT device constraints?
- Is machines holding money safe?
Translating to technical requirements

1. Enable machines to make agreements, handle disputes: use enforceable digital agreements, algorithmic dispute resolution.

2. Design for IoT scale: make the interactions in a transaction mostly peer to peer

3. Design to run on constrained IoT devices: consider limitations on compute, memory, network, power consumption

4. Limit/eliminate the need for machines to hold money: use pre-authorizations
Solution
Perun State Channels

On-Chain (Layer 1)

1. Open/Fund
2. Transact
3. Dispute/Settle

Perun Smart Contract

Off-Chain (Layer 2)

1. Open/Fund
2. Transact
3. Dispute/Settle

Perun channel
Perun IoT State Channels

On-Chain (Layer 1)

1. Open/Fund
2. Transact
3. Dispute/Settle

Perun Smart Contract

Off-Chain (Layer 2)

1. Open/Fund
3. Dispute/Settle

Perun channel

1. Open/Fund
2. Transact
3. Dispute/Settle
Translating to technical requirements

1. Enable machines to make agreements, handle disputes: use enforceable digital agreements, algorithmic dispute resolution.

2. Design for IoT scale: make the interactions in a transaction mostly peer to peer

3. Design to run on constrained IoT devices: consider limitations on compute, memory, network, power consumption

4. Limit/eliminate the need for machines to hold money: use pre-authorizations
Let’s look under the hood
What makes IoT State Channels work?

1. Split the on-chain and off-chain components of perun protocol (Proposal #3).
2. Design a protocol for externalizing watching service (Proposal #4).
3. Use standardized schemes for encoding off-chain messages (go-perun v0.9.0)
4. Use the external funding and watching components (in demo, to be released)
Putting things together
Go-perun

Perun Protocols

- Level DB
- Persistence
- Networking & Serialization

- Ethereum
- Cosmos
- Polkadot
- Internet Computer
- Chain
- TCP
- Perun format
- Protocol Buffers
Perun node

- Use case specific API (e.g: payment)
- Remote interface for API (e.g: gRPC)
- Funding, watching services for IoT light clients

Session
(manage channels, keys/wallets, off-chain IDs of peers)

go-perun

Level DB
Persistence

Perun Protocols

Ethereum Chain

Networking & Serialization
Perun format

TCP
in near future, we plan to focus on

- implement a light client for deep embedded devices (bare metal, RTOS).

and ...  we are looking for contributions!

- find the concept relevant?
- like to evaluate?
- interested in joining our development efforts?
Thank you :)

Open for discussion :)