Trusted D2D-based IoT Resource Access using Smart Contracts

Vasilios A. Siris
joint work with D. Dimopoulos, N. Fotiou, S. Voulgaris, G.C. Polyzos
Mobile Multimedia Laboratory
Athens University of Economics and Business, Greece
vsiris@aueb.gr

IEEE WoWMoM 2019
10-12 June 2019, Washington DC, USA

EU H2020 SOFIE: Secure Open Federation for Internet Everywhere

Motivation and goal

- Why constrained IoT environments ?
- Why (not) blockchains ?
- Goal: investigate options for integrating blockchains with authorization to constrained IoT devices with different cost/functionality tradeoffs
- Key challenges
  - Transaction cost and delay
  - Fully decentralized solution
  - Ensuring that IoT devices actually provide promised access
  - Constrained client devices & constrained resource devices

Addressed in this paper

Single public ledger not enough

Blockchain interaction with real world is a challenge

vsiris@aueb.gr

Why constrained IoT environments?

- Because many IoT devices are constrained in terms of
  - processing and storage resources
  - network connectivity

Reducing usage also ***reduces power consumption*** & ***security threats***

Scalability of IoT systems ***can be addressed*** by utilizing device-to-device communication

Device-to-device technologies ***exist*** and are ***becoming mature***

New challenge: how to achieve ***trusted*** device-to-device communication

vsiris@aueb.gr

Why blockchains? Blockchain features

- **Decentralized trust,** i.e. no single trusted third party
  - Public ledgers: *wide-scale decentralized trust*
  - Permissioned ledgers: *degree of trust* determined by permissioned set
- **Immutability**
  - related to first point, majority of nodes need to agree to change state
- **Transparency**
  - not only a feature but a *requirement* for decentralized trust
  - tradeoff with *privacy*
- **Availability**, through *decentralized storage and execution*
  - can be achieved other ways

vsiris@aueb.gr

Two baseline models

- Immutable recording of transactions and events
  - Cryptographically link authorization grants to blockchain payments
  - Record hashes of authorization messages exchanged on blockchain

Model 1: Authorization grants linked to blockchain payments and hashes recorded

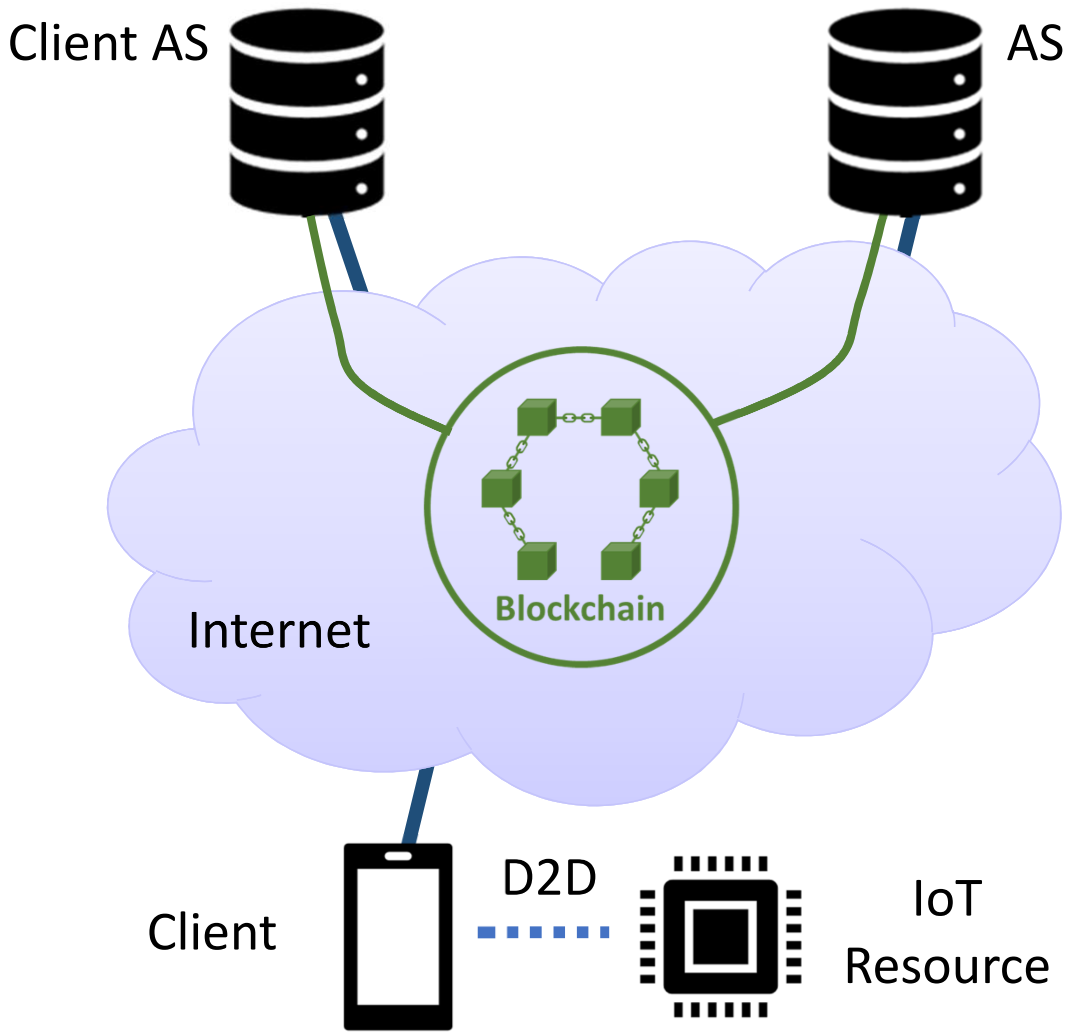
- Transparent and trusted execution of authorization logic
  - More expressive than above
  - Policies can involve IoT events recorded on blockchain
  - Can benefit from blockchain's high availability
  - But more expensive

Model 2: Smart contract handling authorization requests and encoding policies

vsiris@aueb.gr

Assumptions

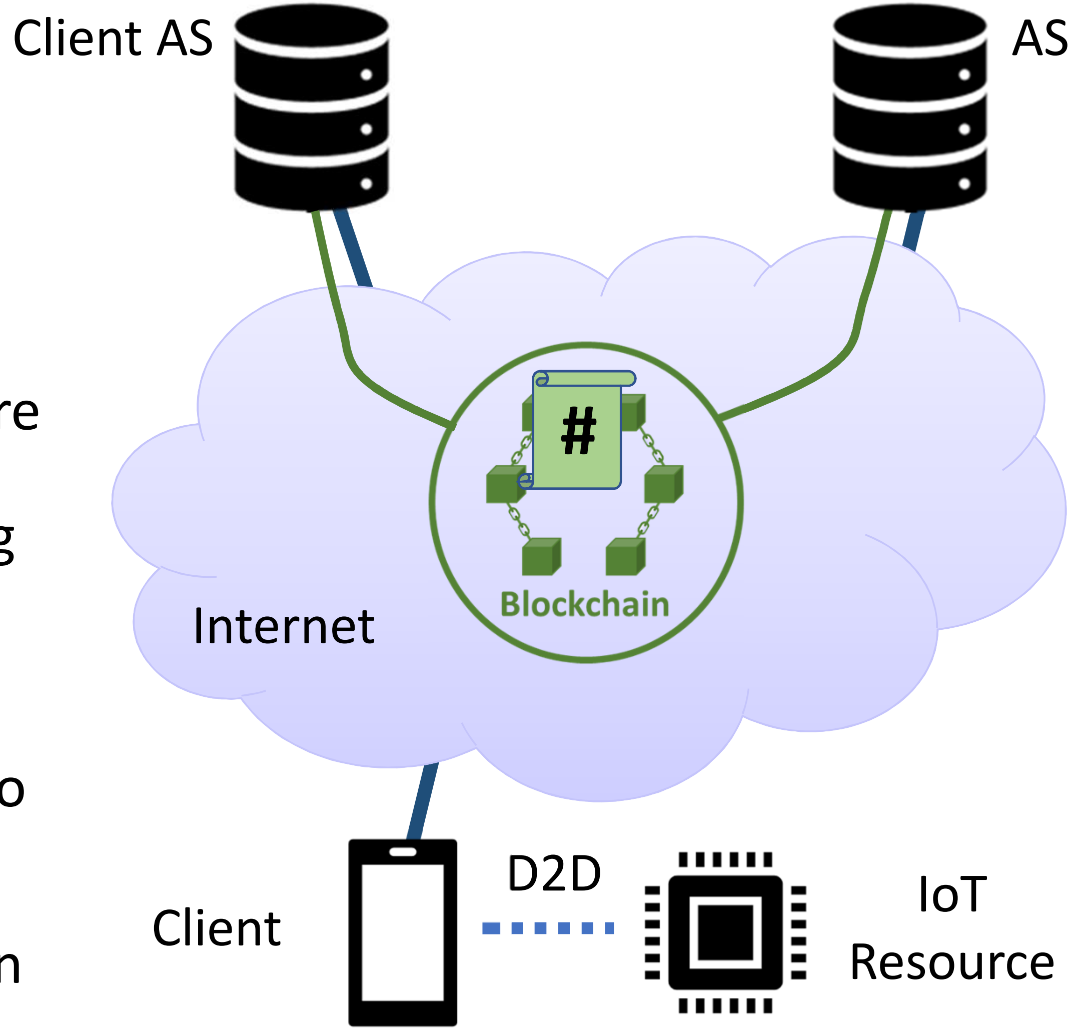
- Client and IoT resource interact only using D2D and can have limited resources and network connectivity
  - *Previous work assumes clients and IoT devices always connected and interact directly with blockchain*
- Authorization Server (AS) handles requests on behalf of IoT resource and client AS handles requests for client
  - OAuth 2.0 authorization framework
  - Based on access tokens
- AS and client AS always connected and can interact with blockchain



vsiris@aueb.gr

Model 1: Authorization grants linked to blockchain payments and hashes recorded

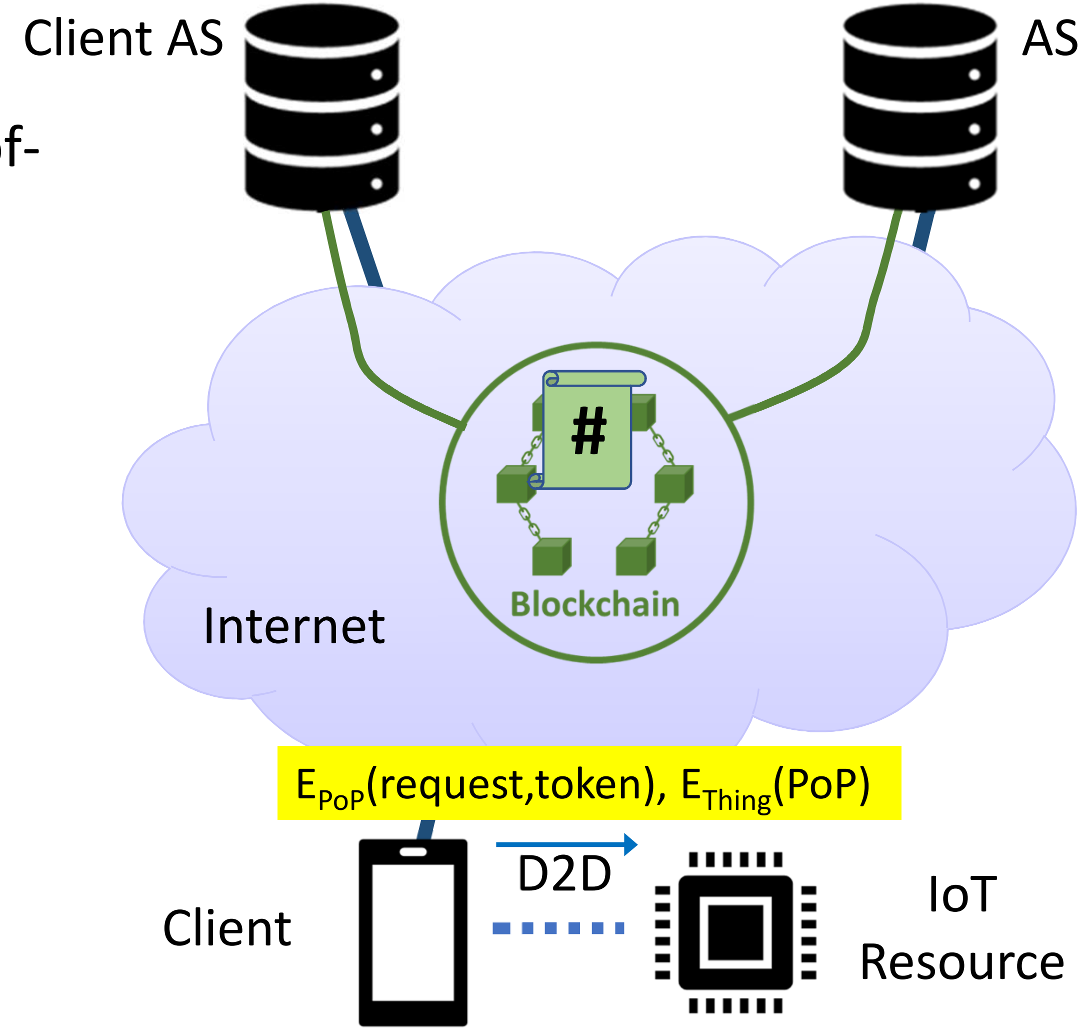
- Client AS and AS communicate directly as in OAuth 2.0
- Access token encrypted with secret s
- Secret s related to payment's hash-lock
- Proof-of-Possession (PoP) used to secure client-IoT resource D2D link
- Client AS deposits amount for accessing resource
- Deposit transferred to resource owner when s revealed on blockchain
- Client AS reads secret s on blockchain to decrypt access token
- Hash of messages exchanged between client AS and AS recorded on blockchain



vsiris@aueb.gr

Model 1: Authorization grants linked to blockchain payments and hashes recorded

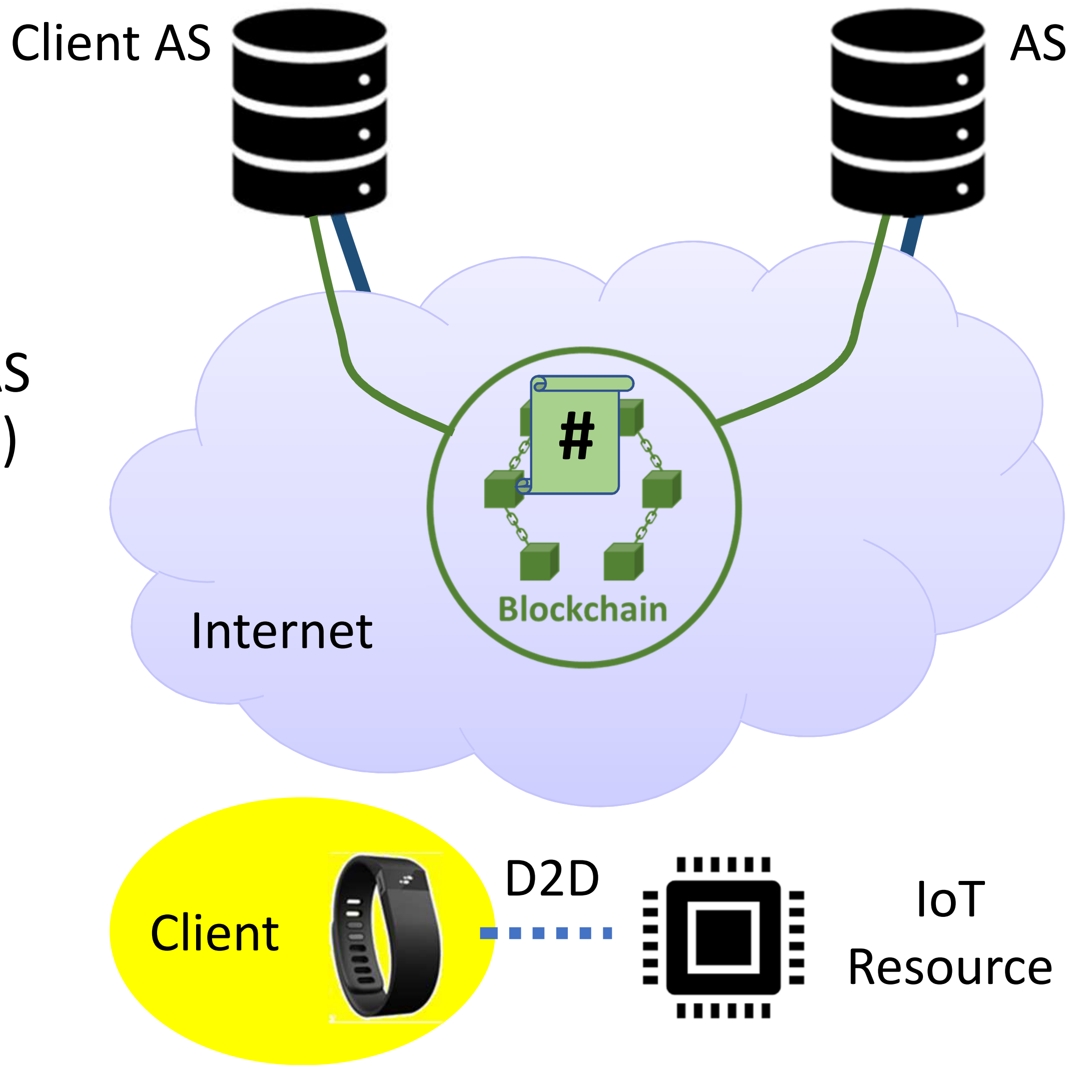
- Client AS sends to client token, Proof-of-Possession (PoP) and encrypted PoP
- PoP used to secure client-IoT resource D2D link



vsiris@aueb.gr

Disconnected resource & ~~connected~~ disconnected client

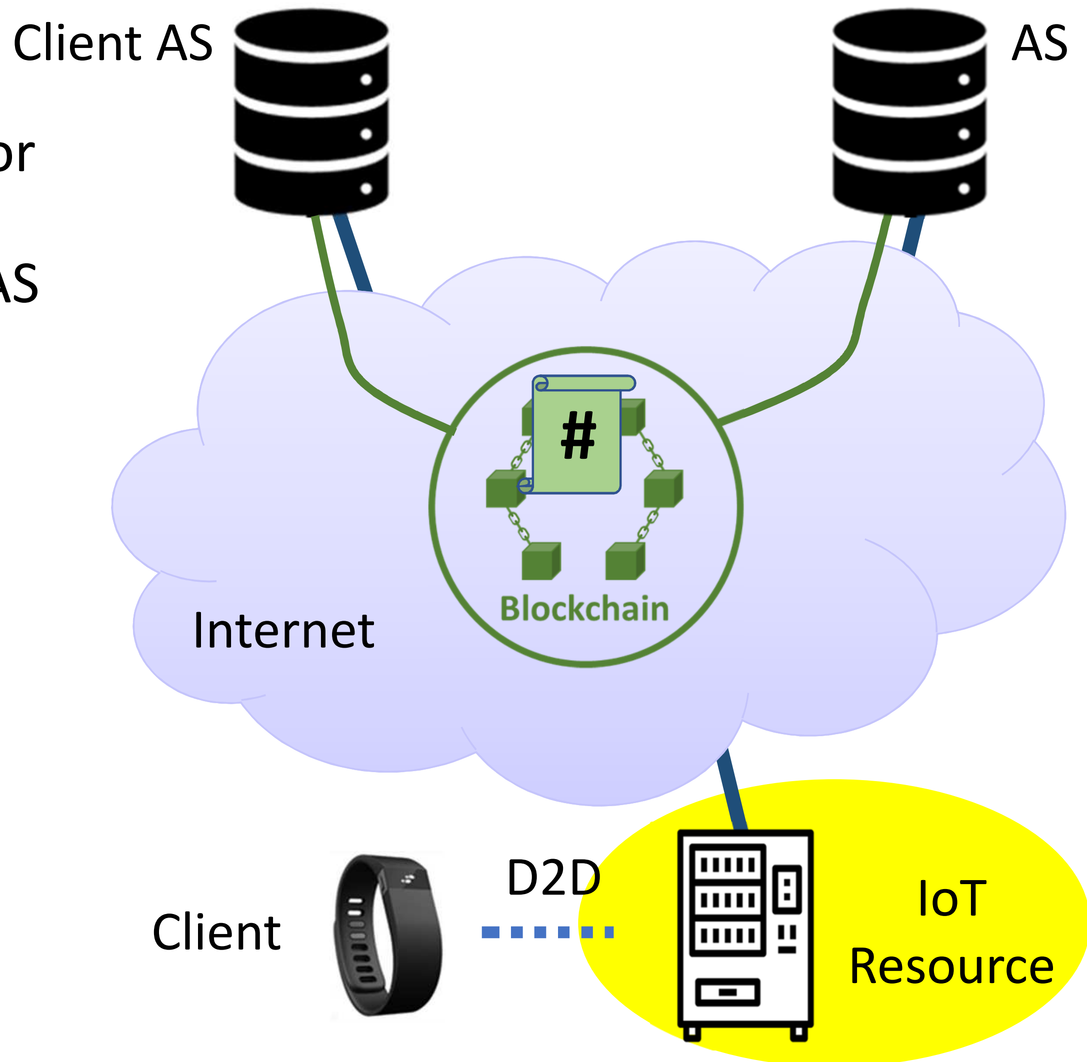
- Up to now assumed that client has continuous connectivity
- Disconnected client: obtains authorization information from client AS at some prior instance (asynchronously)



vsiris@aueb.gr

Connected resource & disconnected client

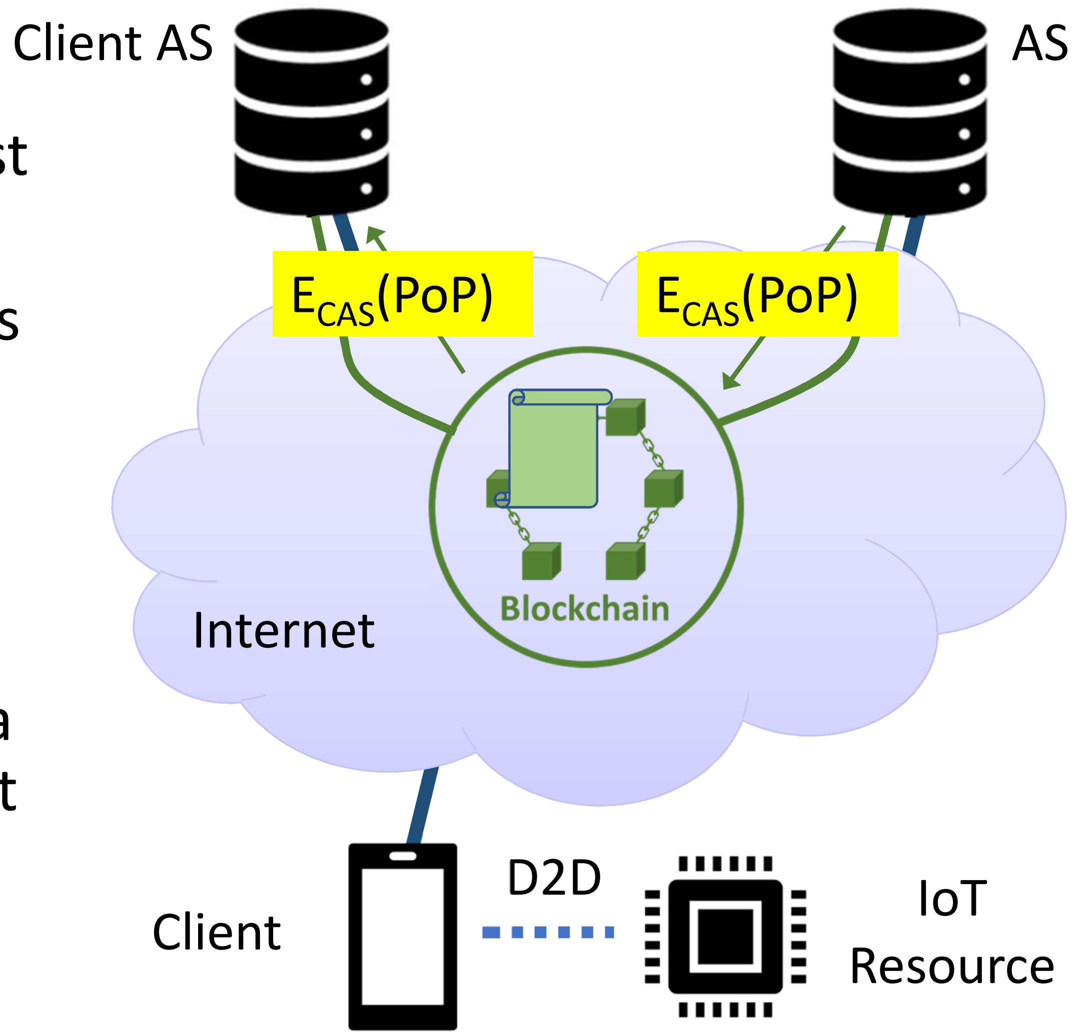
- Connected IoT resource: acts as relay for disconnected client and can obtain authorization information from Client AS synchronously



vsiris@aueb.gr

Model 2: Smart contract handling authorization requests and encoding policies

- Client AS sends authorization request to Smart Contract
- Smart Contract transparently records prices and authorization policies (defined by resource owner)
- As in previous model, payments linked to authorization requests
- Unlike previous model: because data on blockchain public need to encrypt part of token with CAS's public key



vsiris@aueb.gr

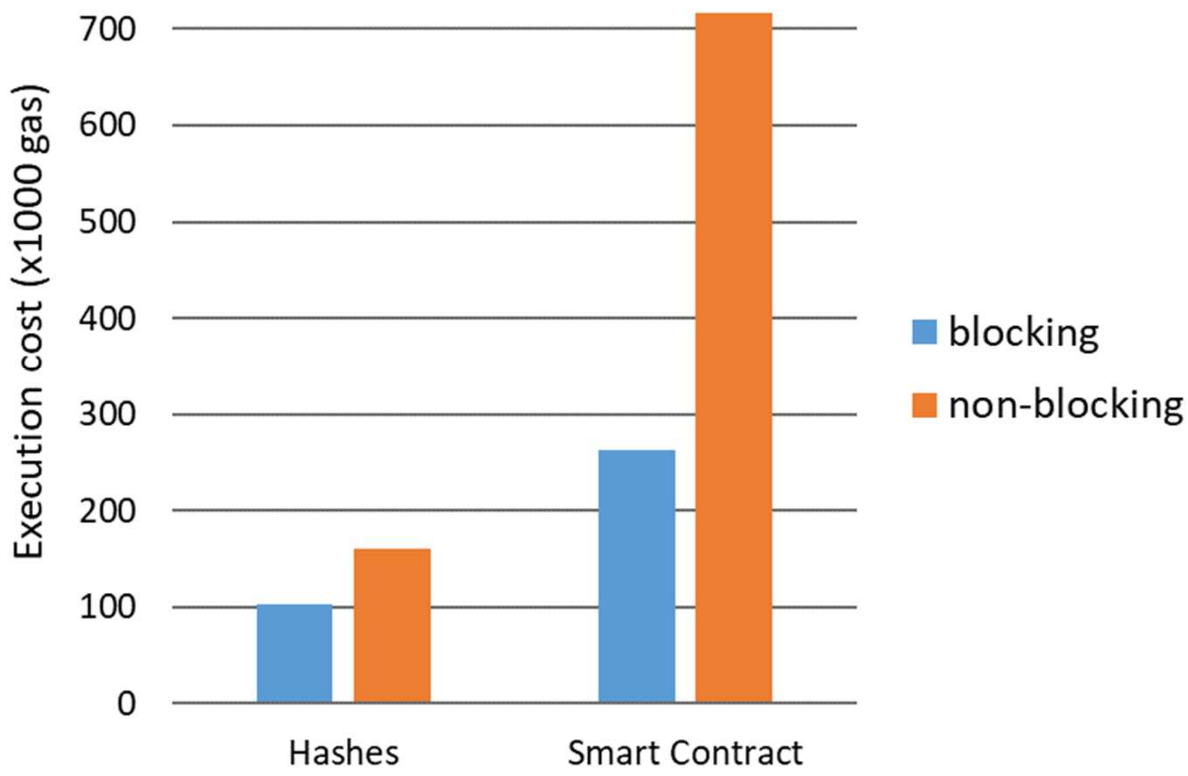
Implementation

- Deployed local node connected to Rinkeby and Ropsten public Ethereum testnets
  - Private chain is a local Ethereum network
- Smart contract written in Solidity with Remix web-based editor
- Web3.0 to interact with Rinkeby and Ropsten blockchains
- Authorization server based on open PHP implementation of OAuth 2.0
- CBOR (Concise Binary Object Representation) Web Token (CWT)
  - More efficient than JSON Web Token (JWT) encoding

vsiris@aueb.gr

Execution cost: blocking & non-blocking

- Smart contract requires ~ 2.5 times EVM gas compared to simply recording hashes
- Only write transactions cost gas
  - Reading data has zero cost
- Non-blocking operation yields higher cost
- Quantifies cost for higher functionality of smart contracts
  - Logic & authorization policies



vsiris@aueb.gr

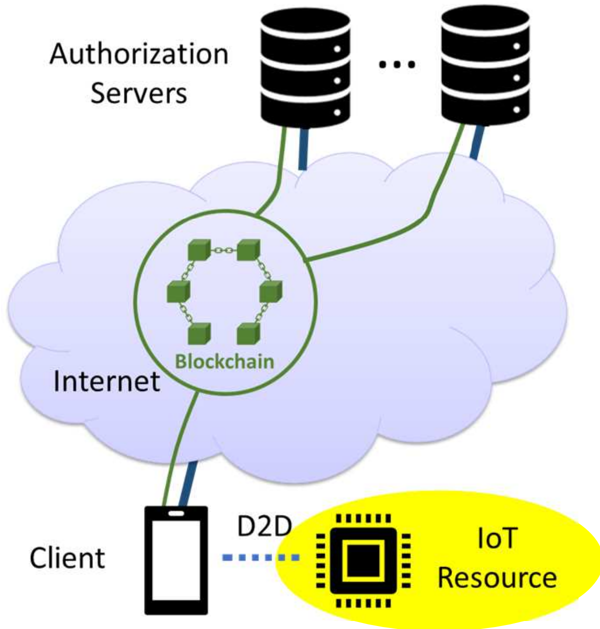
Other challenges

Move smart contract to permissioned ledger and only record hashes on public ledger

Smart contract on public ledger

Achieved by combining public with private/permissioned ledger

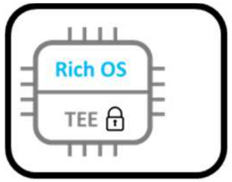
- High cost & delay incurred by blockchains
  - Due to public ledger
  - Combining public & private/permissioned ledgers can provide different tradeoffs of cost, trust, and privacy
  - Off-chain transactions: unidirectional payment channels sufficient for some IoT applications
- Single AS
  - Blockchain advantages are limited to assets & transactions residing in the blockchain
  - Once we traverse blockchain boundaries we loose these benefits
  - Adding multiple ASes not a solution because IoT resource not directly connected to blockchain
  - Need processing at client to reduce data & ensure trust with constrained IoT resource



vsiris@aueb.gr

Other challenges (cont)

- Trust that resource indeed provides access
  - Trusted Execution Environments (TEEs) such as ARM’s TrustZone, Intel’s SGX, Keystone (open source RISC V)



IoT resource
with TEE

Papers – see also https://mm.aueb.gr/blockchains/
“IoT Resource Access utilizing Blockchains and Trusted Execution Environments”, Global IoT Summit 2019
“Secure IoT access at scale using blockchains and smart contracts”, IEEE IoT-SoS 2019
“Trusted D2D-based IoT Resource Access using Smart Contracts”, IEEE WoWMoM 2019
“Smart Contracts for Decentralized Authorization to Constrained Things”, CryBlock 2019 workshop at IEEE INFOCOM 2019
“OAuth 2.0 meets Blockchain for Authorization in Constrained IoT Environments”, IEEE World Forum on IoT 2019
“Enabling Decentralised Identifiers and Verifiable Credentials for Constrained Internet-of-Things Devices using OAuth-based Delegation”, DISS workshop at NDSS 2019
“Bridging the Cyber and Physical Worlds using Blockchains and Smart Contracts”, DISS workshop at NDSS 2019
“Interacting with the Internet of Things Using Smart Contracts and Blockchain Technologies”, SpaCCS 2018

vsiris@aueb.gr