DISTRIBUTED RESEARCH ON EMERGING APPLICATIONS & MACHINES
Department of Computational & Data Sciences
Indian Institute of Science, Bangalore

DREAM:Lab

A Distributed and Decentralized IoT Data Exchange

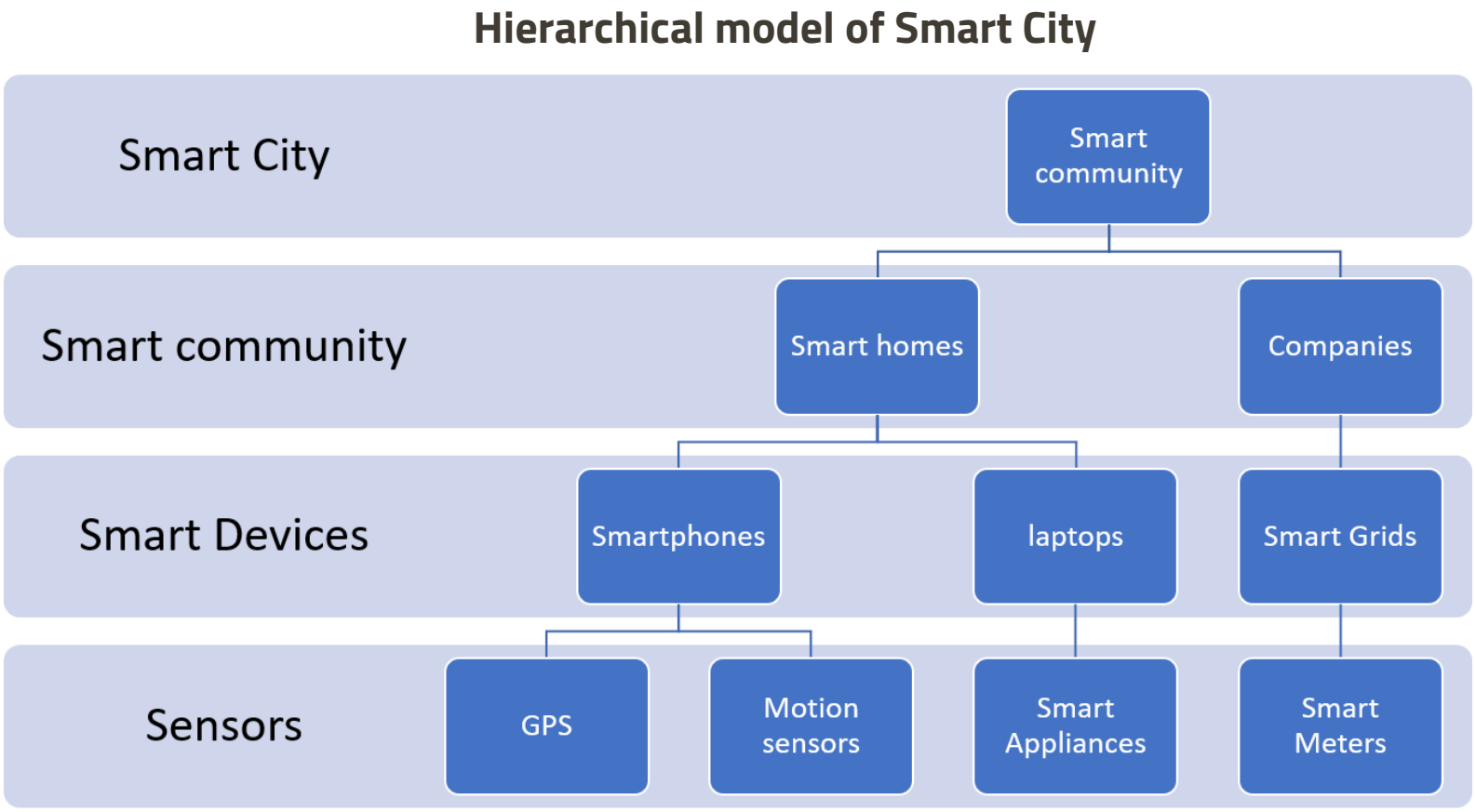
Shrey Baheti
Advisor: Yogesh Simmhan

Introduction

Motivation

- Billions of devices, sensors generating petabytes of IoT data daily.
  - By smart cities, communities, hobbyists (B2B, B2C, C2C)
  - Held privately, unavailable for open exchange.
  - Costs, incentives for sharing data.
  - Trust in data?
- Data is only valuable if it can be processed.
  - Fast Data is perishable. Timeliness.
  - Integrate diverse data sources for decisions.
- Need a common mechanism to exchange distributed IoT data for benefits (money, attribution) in a trustworthy and scalable manner.

Application Scenario



Background

Different Approaches

Centralized Data Exchange

- Relies on brokered communication models (client/server paradigm).
- Requires huge processing and storage capacities (governed by large organizations).
- Requires trusted third party to govern access control policy (Intermediary).
- Results in bottleneck and single point of failure.

P2P Data Exchange

- Direct communication between peers.
- Huge reduction in processing and storage requirements.
- Data is distributed and shared among peers, no access control (e.g. Bittorrent).
- No need of trusted third party.
- No audit trail of consumption and ownership of data.

Data Exchange using Distributed Ledger

- Distributed Ledgers
  - Database spread across peers.
  - Peer replicates and maintain identical copy.
  - No central authority, govern by consensus.
  - Will work as long as majority of peers are honest.

Data Exchange using Distributed Ledger

- Blockchain Technology
  - Blockchain is form of distributed ledger.
  - A hash pointer append-only data structure of blocks.
  - Altering or deleting previously entered data is provably impossible
  - Trust through immutable, time-stamped transactions.
  - Transparent since identical copy of ledger is maintained by everyone.

Data Exchange using Distributed Ledger

- Blockchain Types

	Public	Consortium	Private
Consensus	All miners	Selected set of nodes	One organization
Read Permission	Public	Public or restricted	Public or restricted.
Efficiency	Low	High	High
Data Security	Very High	Data Breach(plausible)	High
Example	Bitcoin Ethereum	R3 conda, B3i	Multichain

Problem Formulation

Problem Description

- Objective
  - Device Owners can share data collected by IoT devices among interested parties for benefits.
  - Build a secure, trusted, transparent and access controlled IoT data exchange.
  - Providing necessary access control and auditing over data exchanged.

How can we securely exchange data for benefits?

- Goal
  - Atomic transaction to securely transfer of data for benefits.
- Challenges
  - Lack of security protocols standardization.
  - Dependence on trusted third party.
- Approach
  - Blockchain based smart contract (more on this later) for access control (decryption key shared) and payment.
  - Storing metadata and full encrypted data on distributed file storage (IPFS).

How the data owner claims ownership of data generated from sensor?

- Goal
  - Entity that owns the IoT device also owns the data produced by that device
- Challenges
  - Transfer of ownership on purchase of data
  - Possession of data ≠ Ownership
- Approach
  - Hash of sensor id and metadata recorded on immutable distributed ledger.
  - Assuming device is owned by single party.

How can we establish the provenance of derived data for authenticity?

- Goal
  - Tracking of the journey of data consumed by multiple parties to authenticate.
- Challenges
  - Consumers can derive insights based on data they purchased and resale.
- Approach
  - Transactions recorded on distributed ledger are immutable.
  - Transactions are non-repudiable through cryptocurrency.

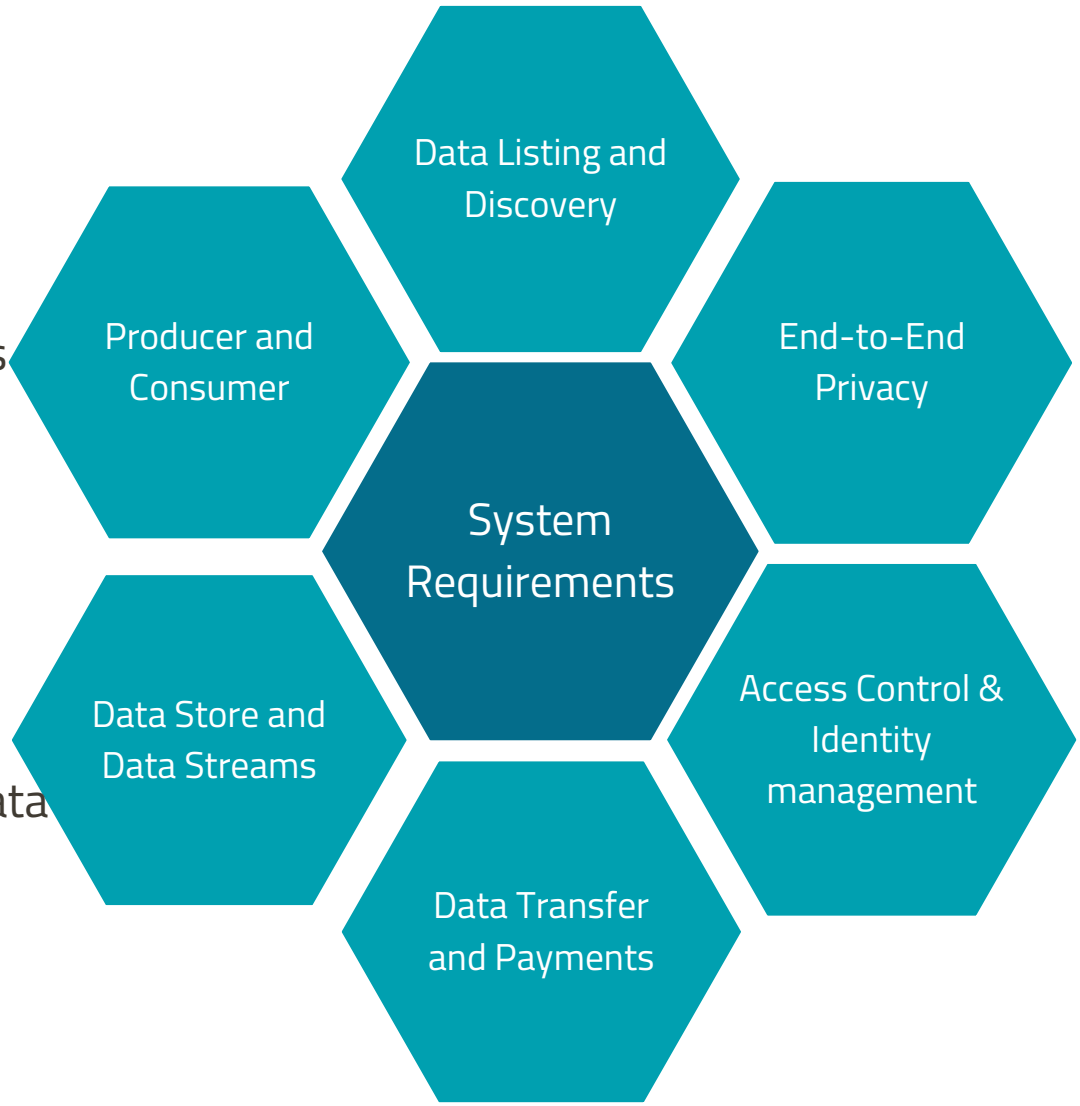
How can we support streaming data, with freshness and scaling?

- Goal
  - Timely delivery of data
- Challenges
  - Time critical applications have real-time constraints
- Approach
  - Smart contracts to enable micropayments (work in progress) to subscribe streaming data

Design Requirements

System and Security

Data offers visible to everyone



Willing to share data for benefits

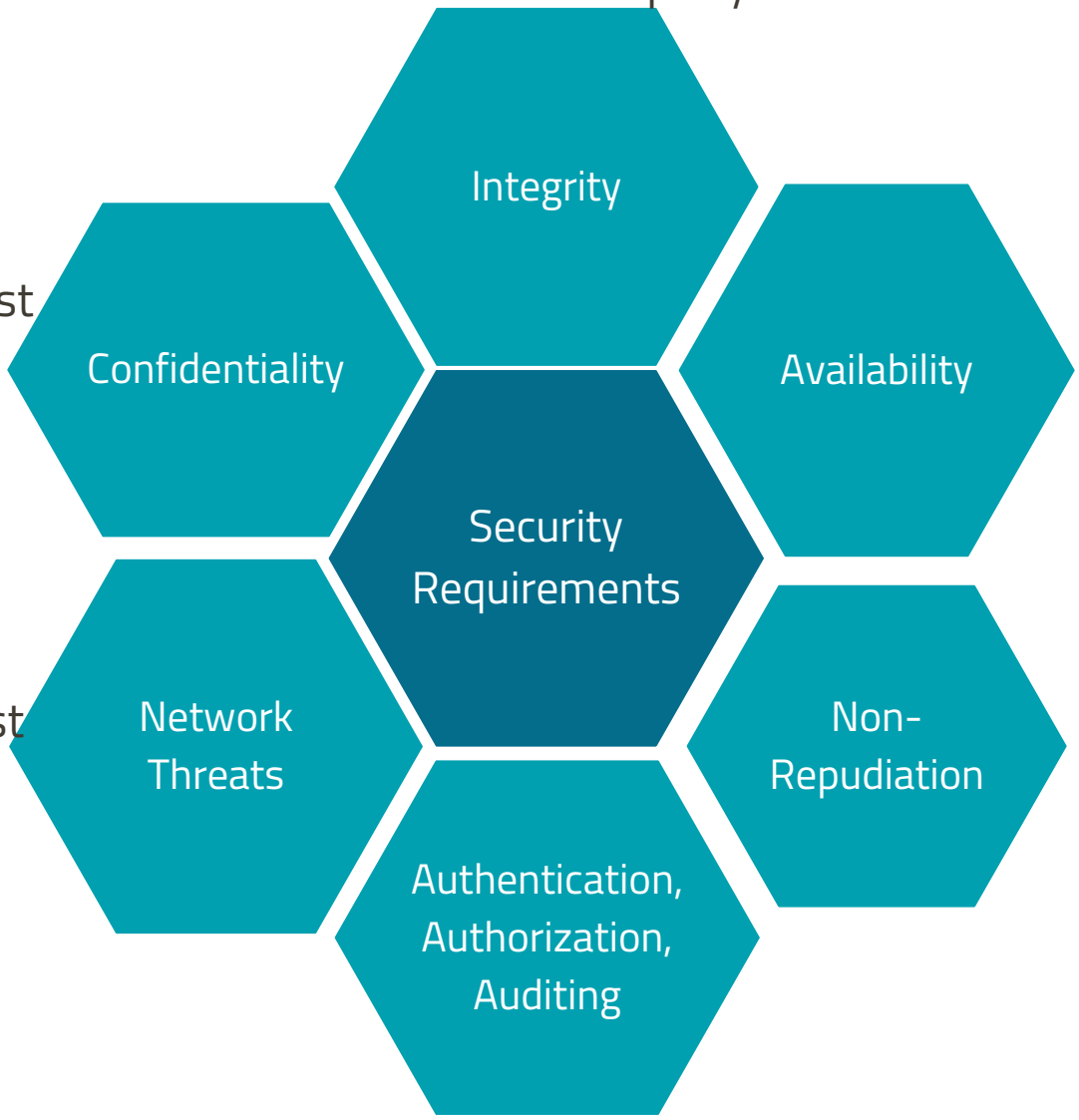
Sensitive data is not logged/store on to public ledger/exchange

Local storage and common transfer mechanism for data

Fine-grained access control to grant/revoke permissions

Atomic transaction to transfer access key in exchange of payment

Data is unaltered – except by owners



Prevention against unauthorized access to data

Service always active for legitimate users

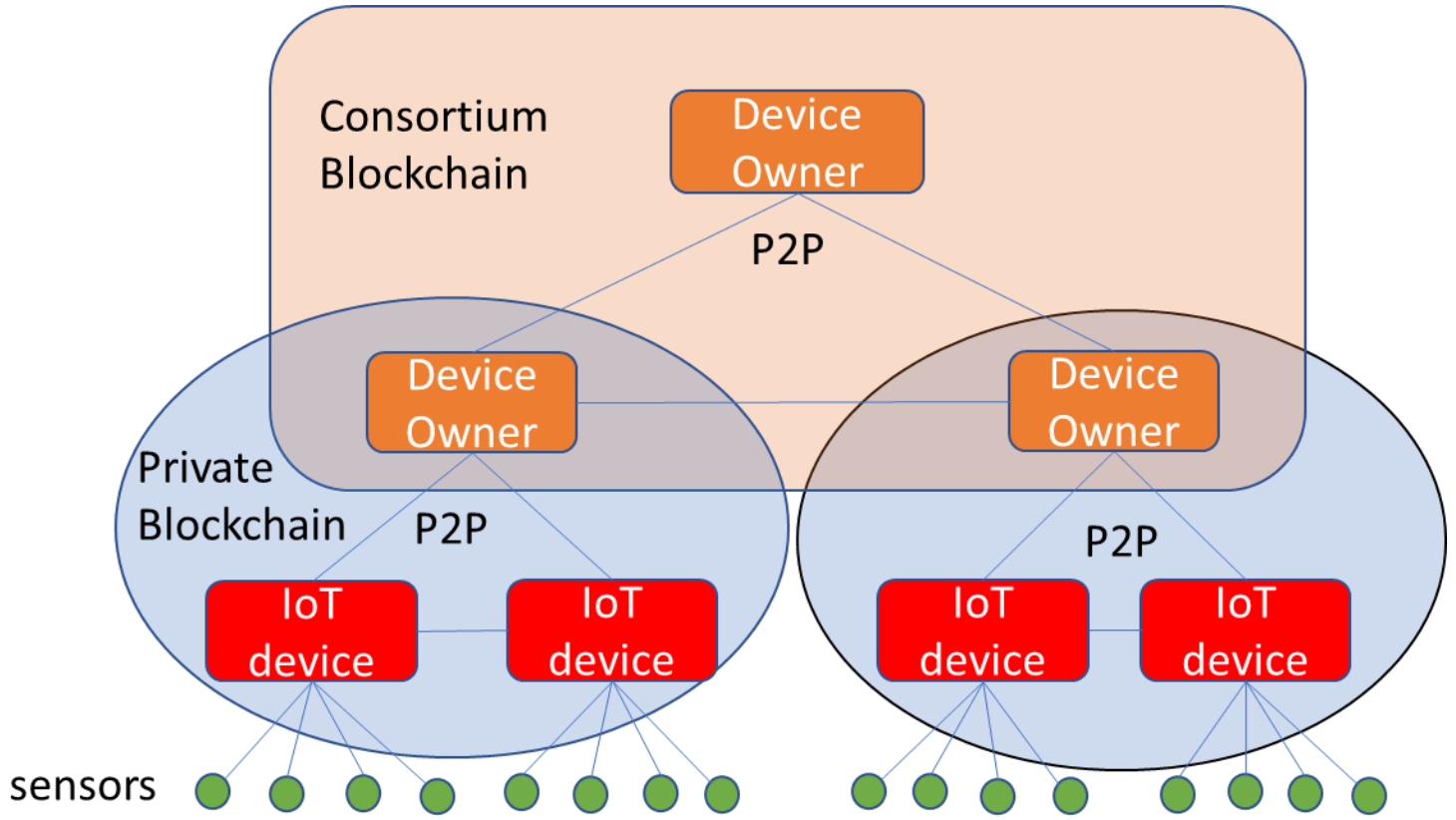
Protection against common attacks

Guarantees against action denial by user

Verify identity & privileges of user and store usage information

System Design

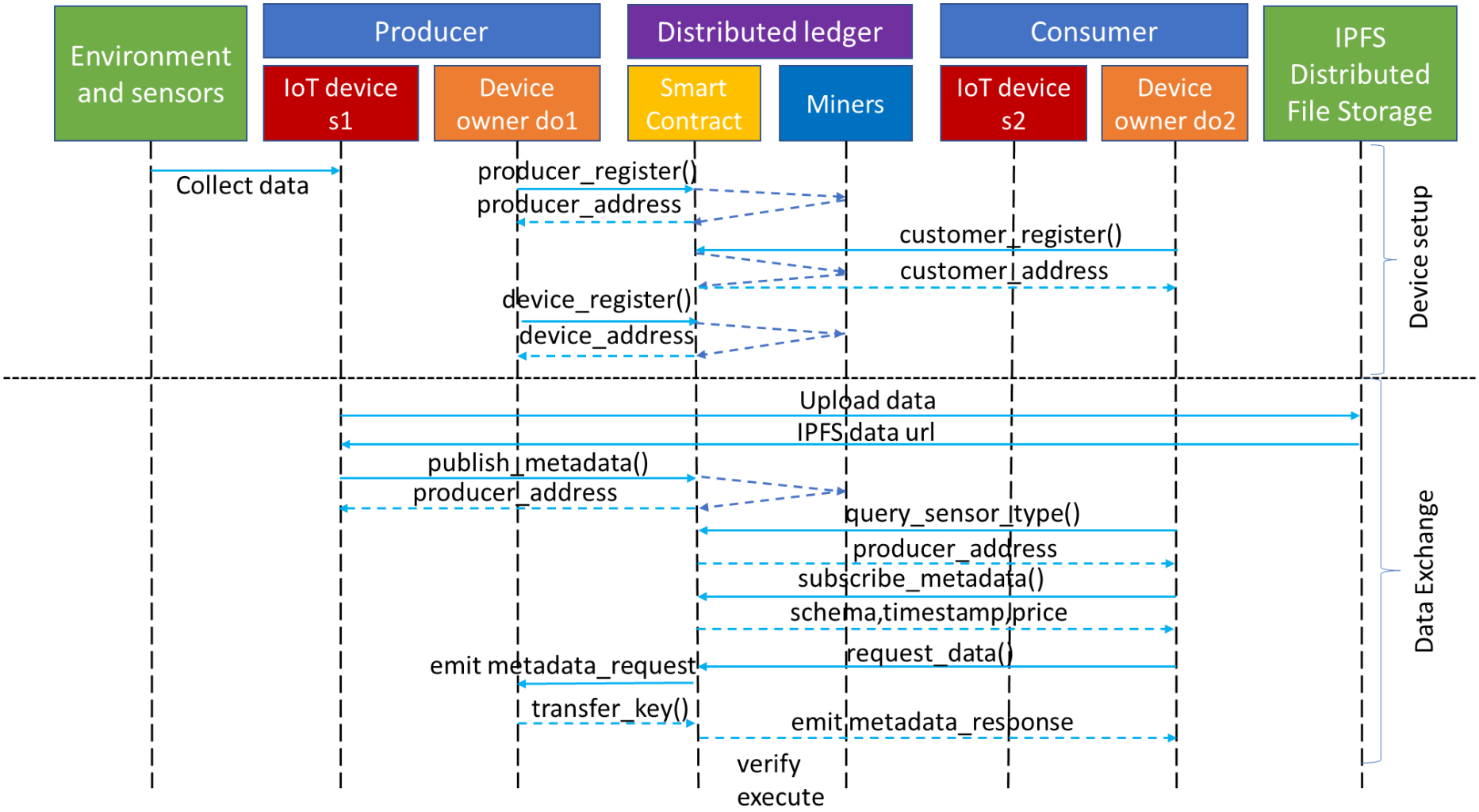
High level overview architecture



Entity and their roles

- Entities
  - **Device Owners** can act as producer and/or consumer.
  - **IoT devices** collects data generated by sensors
  - **Sensors** generating raw data

Sequence Diagram



Evaluation

- Smart Contract Efficiency (deploy = 4,084,450 gas)

Functions	Parameters	Return	Gas Used
producer_register	name, sensors[], costs[]	address	230,468
consumer_register	pub_key	address	572,758
device_register	device_address	address	65,260
publish_metadata	producer_address, sensor_type, schema, timestamp, ipfs, _encID	producer_address	369,324
update_sensor_price	sensor_type, price	Price	33,610
request_data	producer_address, sensor_type, index	emit metadata_request	44,362
transfer_key	dec_key, to, sensor_type, index	emit metadata_response	1,505,852

Cost = gasPrice * gas used *current gasPrice = 10Gwei 1Gwei = 10^{-9} ether

Preparatory Work

VIoLET: A Large-scale Virtual Environment for Internet of Things

Dynamism

- CPU Dynamism
  - variability in the CPU speed
- Used docker's api to update the CPU for a container to enact the change.

Algorithm 1 CPU Dynamism Algorithm

1: δ: fraction of CPU drop
2: ϵ: control interval
3: **procedure** ACTION(*devices*,*p*)
4: **for** d in devices **do**
5: ω: theoretical CPU limit of d
6: π: variability period of d
7: $p \leftarrow \frac{\epsilon}{\pi}$ ▷ $\epsilon << \pi$
8: $r \leftarrow random.uniform(0, 1)$
9: **if** $r <= p$ **then**
10: $\omega^{'} \leftarrow random.uniform((1 - \delta)\omega, \omega)$
11: **end if**
12: **end for**
13: **end procedure**

Dynamism

- Device Reliability
  - devices are transient in nature.
- MTTF
  - expected avg to fail.
- MTTR
  - expected avg time to recover.

Algorithm 2 Device Reliability Algorithm

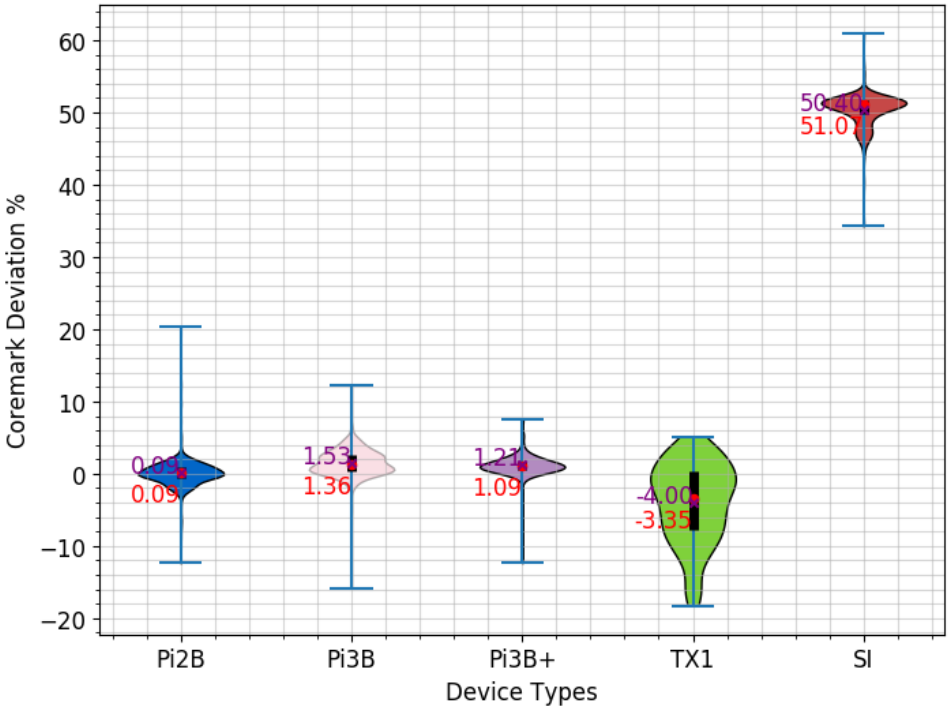
```
1: ϵ: control interval
2: d_n : subset of devices up
3: d_m : subset of devices down
4: procedure DEVICES UP(d_n,p)
5:     for d in d_n do
6:         μ: Mean-Time-to-Failure (MTTF) of d
7:         p ← ϵ/μ                                   ▷ ϵ << μ
8:         r ← random.uniform(0,1)
9:         if r <= p then
10:            d_m ← d
11:            d is stopped
12:        end if
13:    end for
14: end procedure
15: procedure DEVICES DOWN(d_m,p)
16:    for d in d_m do
17:        ρ: Mean-Time-to-Recover (MTTR) of d
18:        p ← ϵ/ρ                                   ▷ ϵ << ρ
19:        r ← random.uniform(0,1)
20:        if r <= p then
21:            d_n ← d
22:            d is started
23:        end if
24:    end for
25: end procedure
```

Experimental setup

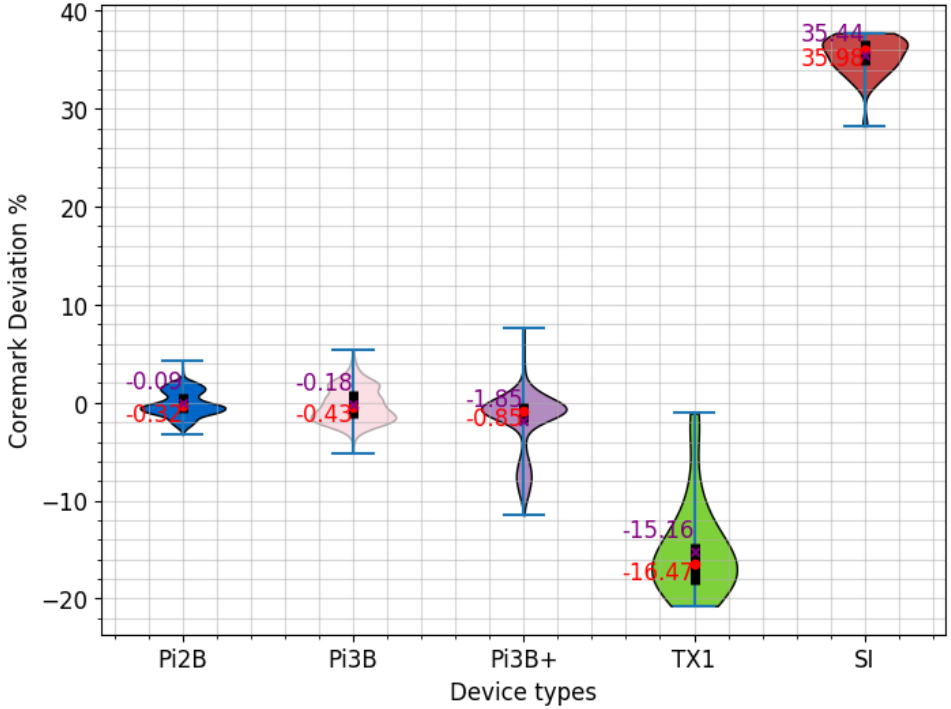
- VIoLET is used to deploy D400 (400 devices) deployment to run dynamism
- CPU Dynamism
  - Variability period = 600 sec
  - Fraction = 0.2
- Reliability
  - MTTF = 600 sec
  - MTTR = 300 sec
  - Control interval = 30 sec
- Duration = 2 hours

Deployment→	**D400**		
Device	Count	∑CM(k)	∑Mem(GB)
Pi 2B	255	2,947	255
Pi 3B	96	1,484	96
Pi 3B+	47	841	47
NVidia TX1	1	27	4
Softiron	1	78	16
Total	**400**	**5377**	**418**
Standard_D32_v3 *(host)*	20	6,025	16,000

Results

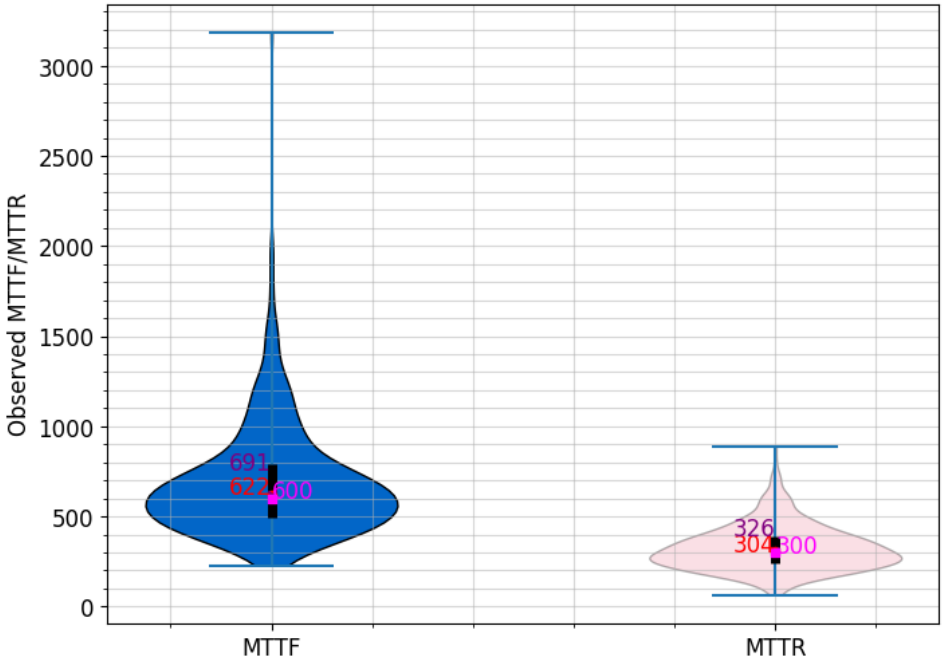


Coremark Deviation **with** CPU dynamism

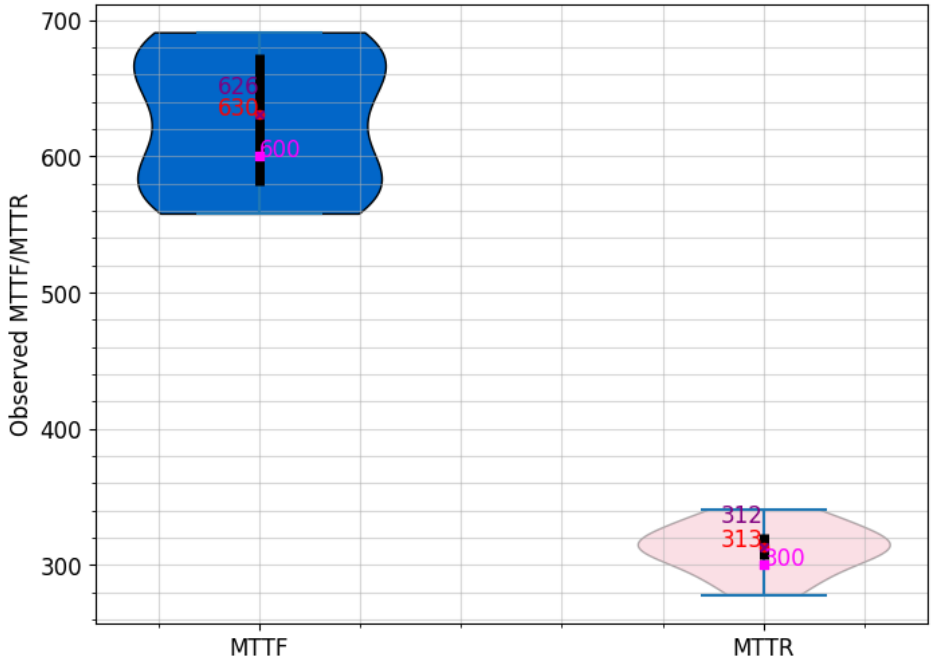


Coremark Deviation **without** CPU dynamism

Results



Across time per device



Across Devices per Interval

Conclusion

- Proof-of-concept IoT data exchange is implemented as a smart contract on Ethereum and uses IPFS as distributed file storage.
- Consumer pub_key (AES,DES) used for confidentiality.
- Exchange does not provide encryption on IoT data stored on IPFS.

Future Work

- Performance evaluation of the above smart contract on actual deployment and VIoLET.
- Support for streaming data via micropayments.
- Interoperability over multiple private networks for IoT data exchange.

References

- S. Badiger, S. Baheti, and Y. Simmhan, "*Violet: A large-scale virtual environment for internet of things*," in Euro-Par 2018: Parallel Processing, M. Aldinucci, L. Padovani, and M. Torquati, Eds. Cham: Springer International Publishing, 2018, pp. 309–324.
- Centralized Approaches
  - K. Miura and M. agar, "*Data marketplace for internet of things*," in 2016 International Conference on Smart Systems and Technologies (SST), Oct 2016, pp. 255–260.
  - B. Krishnamachari, J. Power, S. H. Kim, and C. Shahabi, "*I3: An iot marketplace for smart communities*," in Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services. ACM, 2018, pp. 498–499.
- Decentralized Approaches
  - Ramachandran, G. S., Radhakrishnan, R., & Krishnamachari, B. *Towards a Decentralized Data Marketplace for Smart Cities*.
  - P. Missier, S. Bajoudah, A. Capossele, A. Gaglione, and M. Nati, "*Mind my value: A decentralized infrastructure for fair and trusted iot data trading*," in Proceedings of the Seventh International Conference on the Internet of Things, ser. IoT '17. New York, NY, USA: ACM, 2017, pp. 15:1–15:8. [Online]. Available: http: //doi.acm.org/10.1145/3131542.3131564
- Survey
  - E. F. Jesus, V. R. Chicarino, C. V. de Albuquerque, and A. A. d. A. Rocha, "*A survey of how to use blockchain to secure internet of things and the stalker attack*," Security and Communication Networks, vol. 2018, 2018.

Any Questions?

Smart contract functions

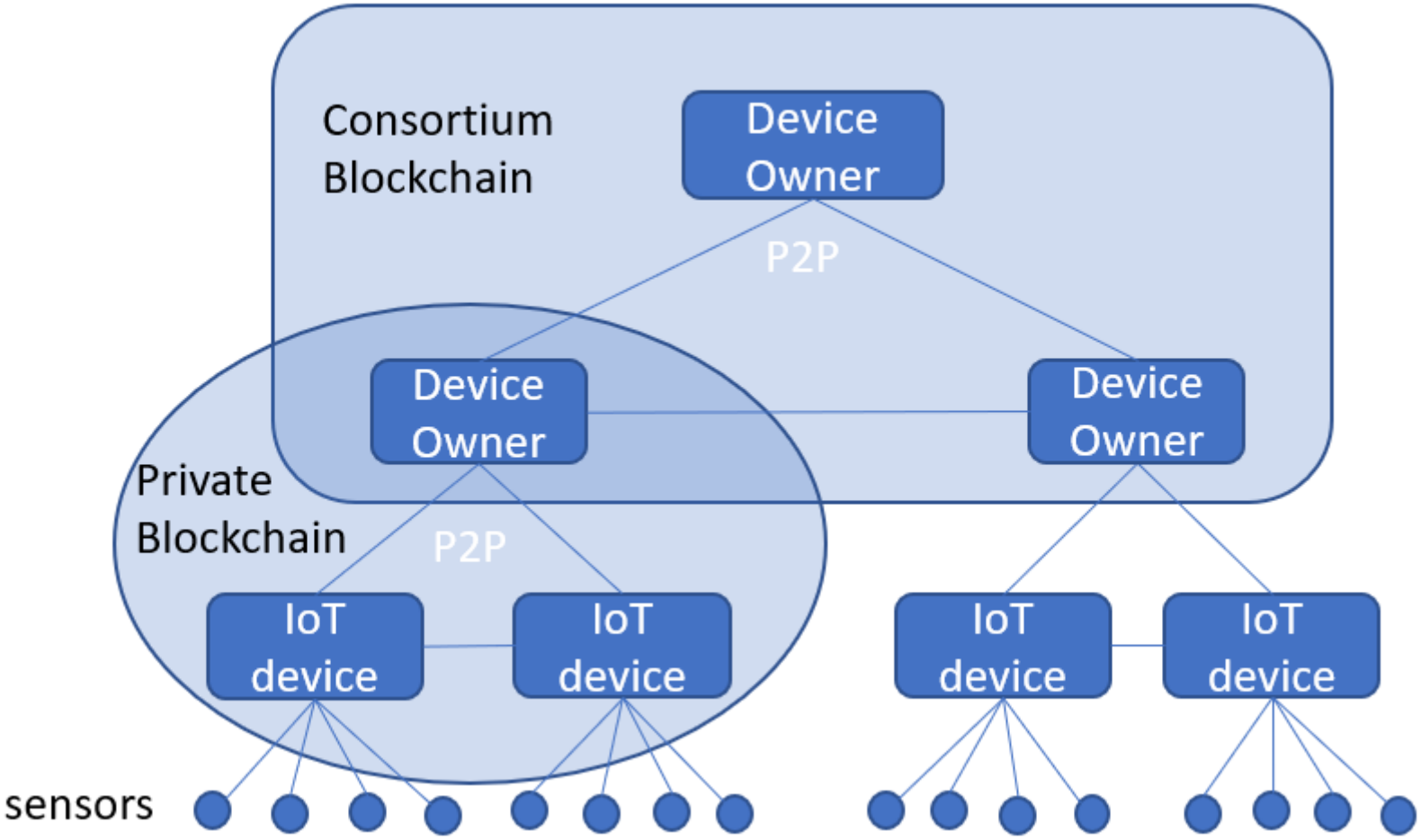
- producer_register(string name, int[] sensors, int[] costs)
  - returns address
- customer_register(string pub_key)
  - returns address
- device_register(address dev_addr)
  - returns address
- get_producer(address addr)
  - returns (string name)

Sensor's metadata functions

- query_sensor_type(int sensor_type, int index)
  - returns (address[] producer)
- publish_metadata(address producer_address, int sensor_type, string schema, int timestamp, string ipfs, int key_index, int enc_id)
  - returns address
- sensor_metadata(address producer_address, int sensor_type, int index)
  - returns (string schema, int timestamp, int price)

Access control and payment functions

- request_data(address producer_address, int sensor_type, int index)
  - emit metadata_request
  - returns producer_address
- transfer_key (string dec_key, address _to, int sensor_type, int index)
  - emit metadata_response
  - returns (string dec_key)


Consortium
Blockchain
Device
Owner
P2P
Device
Owner
Device
Owner
Private
Blockchain
P2P
IoT
device
IoT
device
IoT
device
IoT
device
sensors