Encrypted Databases for Differential Privacy

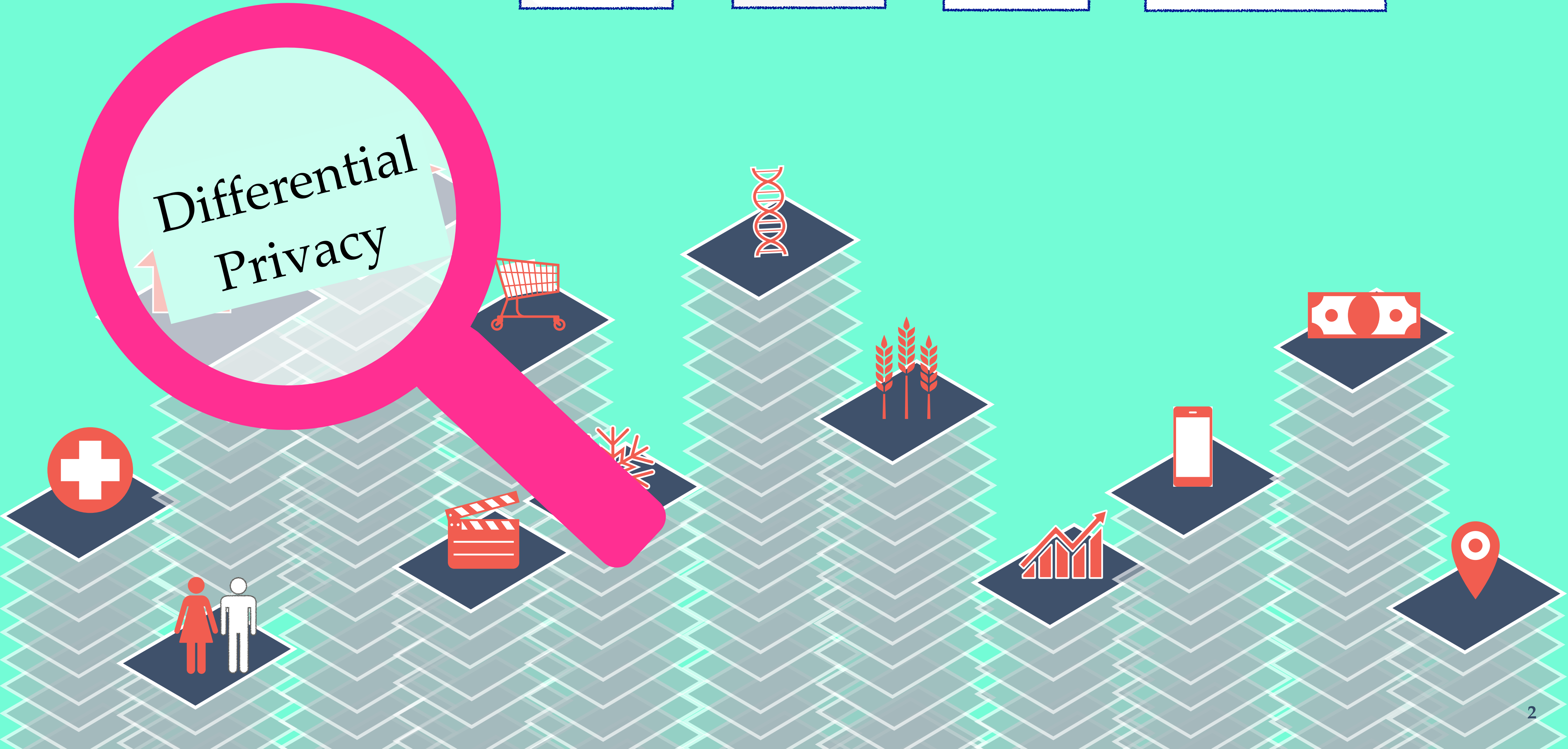
Archita Agarwal, Maurice Herlihy, Seny Kamara, Tarik Moataz

BROWN

ENCRYPTED SYSTEMS LAB

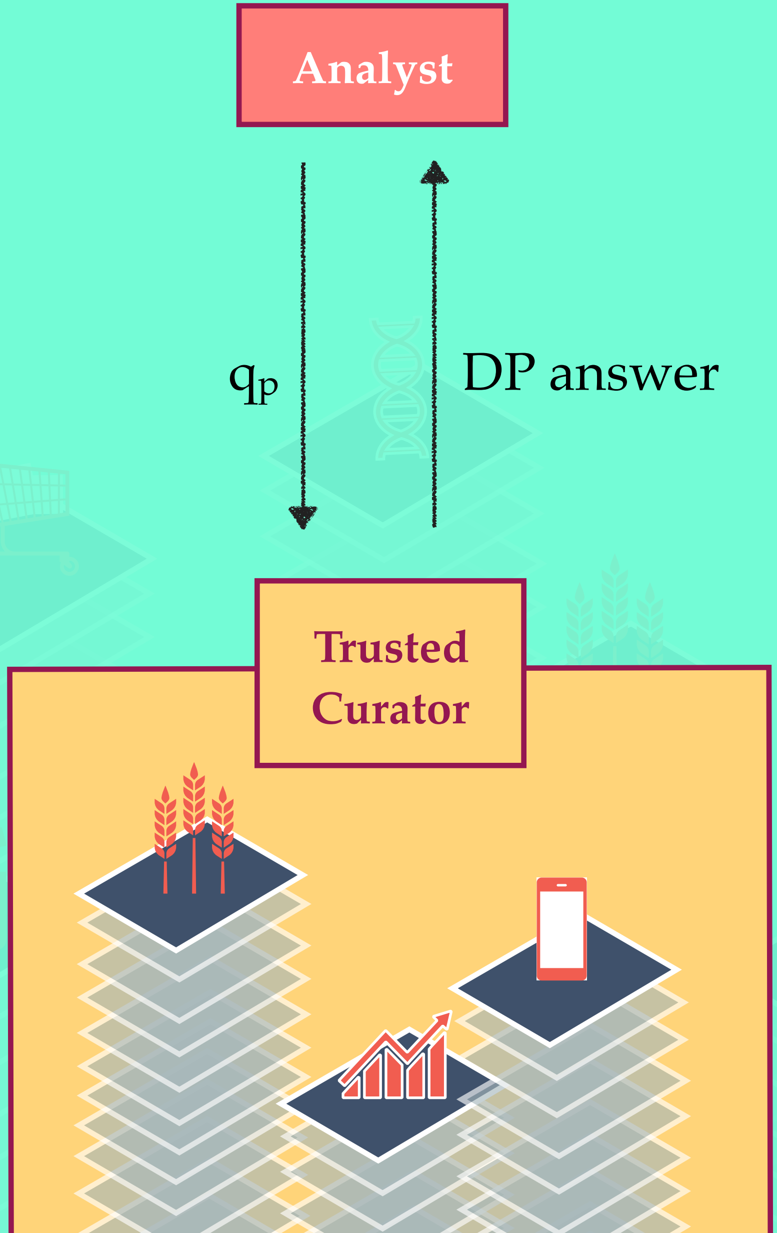
UBER

United States Census Bureau



[DMNS '06]

Differential Privacy



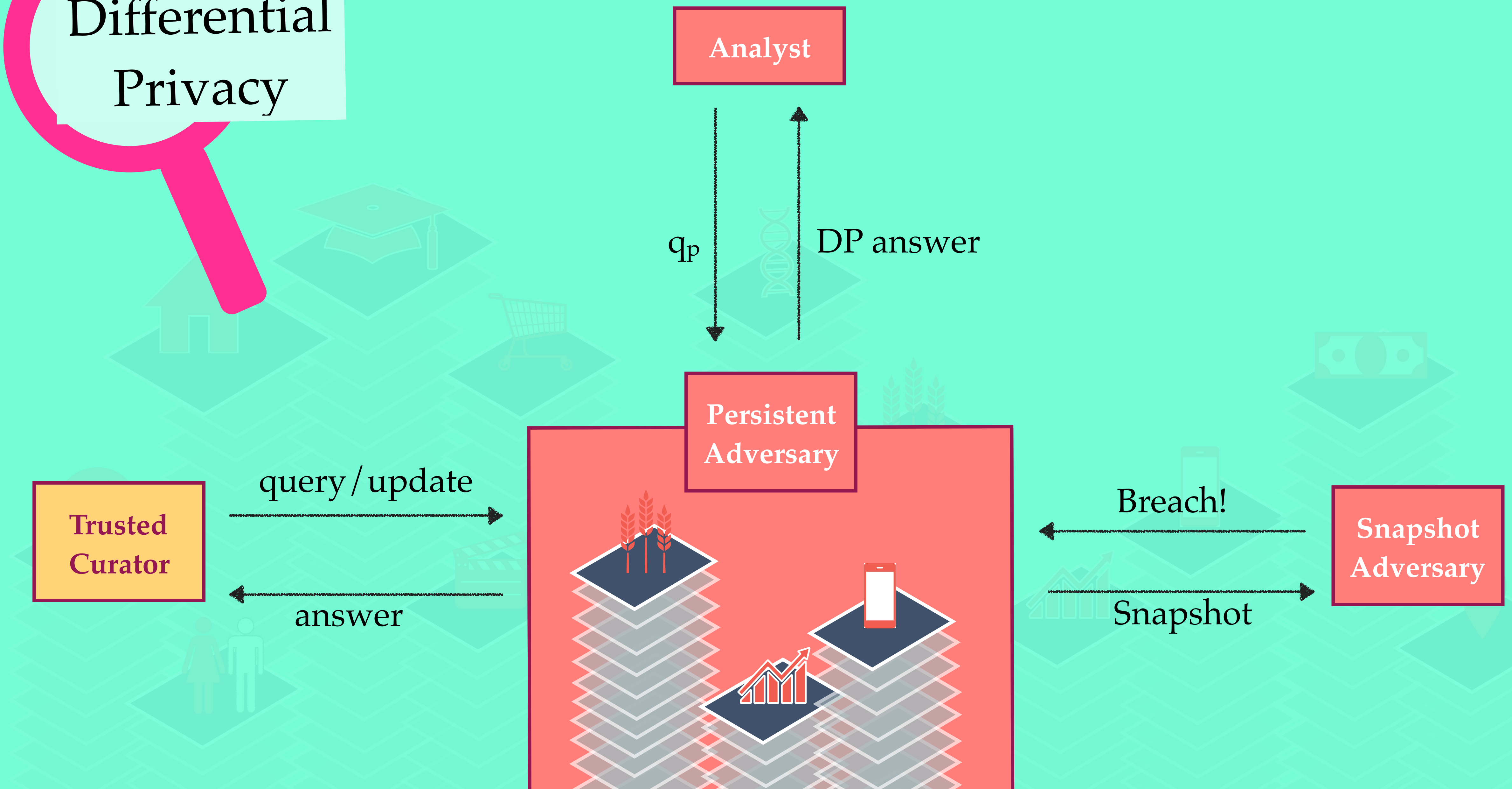
Analyst

q_p

DP answer

Trusted Curator

Differential Privacy



Analyst

q_P

DP answer

Persistent Adversary

Trusted Curator

query/update

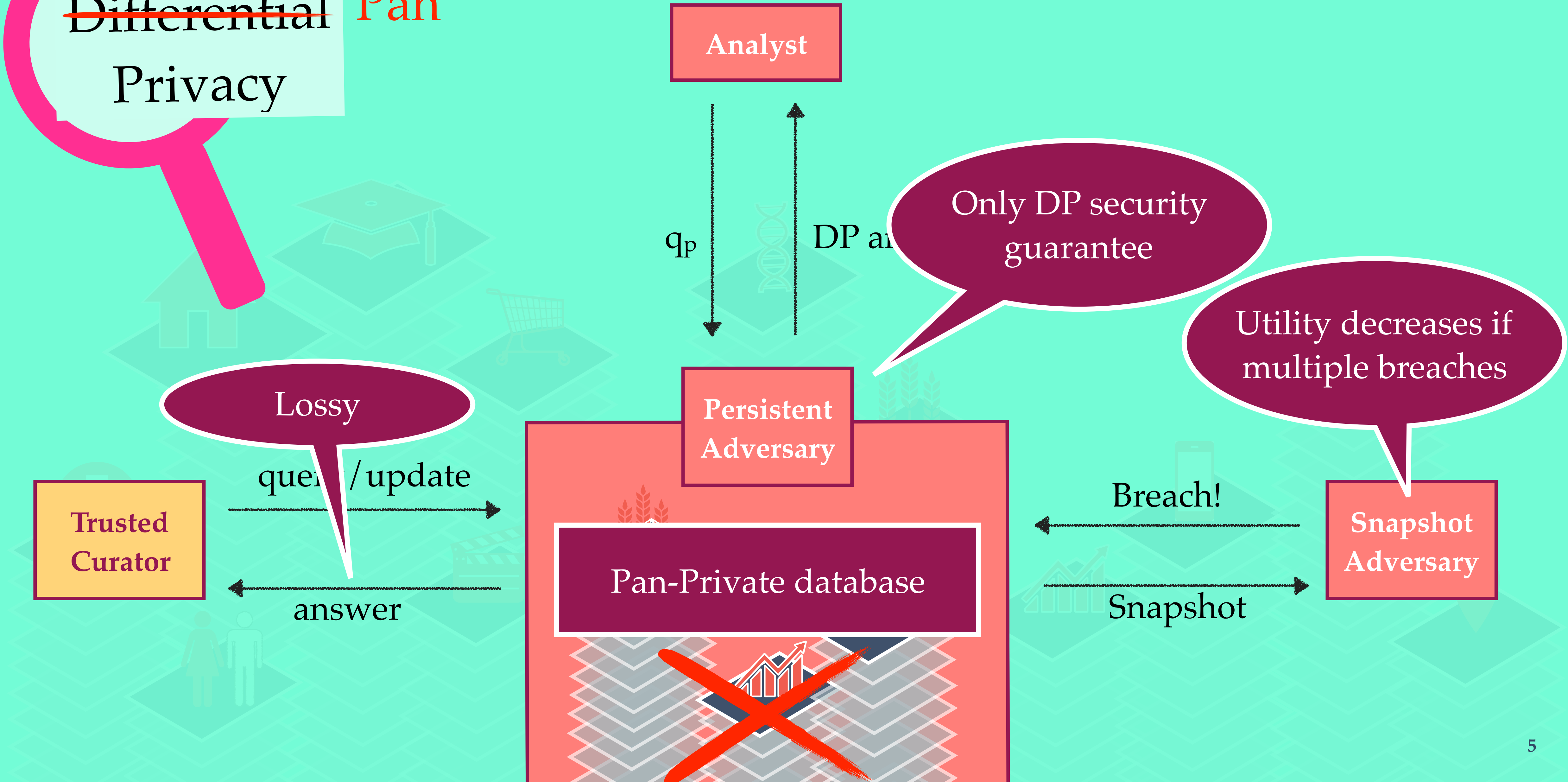
answer

Breach!

Snapshot

Snapshot Adversary

~~Differential~~ Pan Privacy



Analyst

qp

DP ar

Only DP security guarantee

Utility decreases if multiple breaches

Lossy

quer /update

answer

Trusted Curator

Persistent Adversary

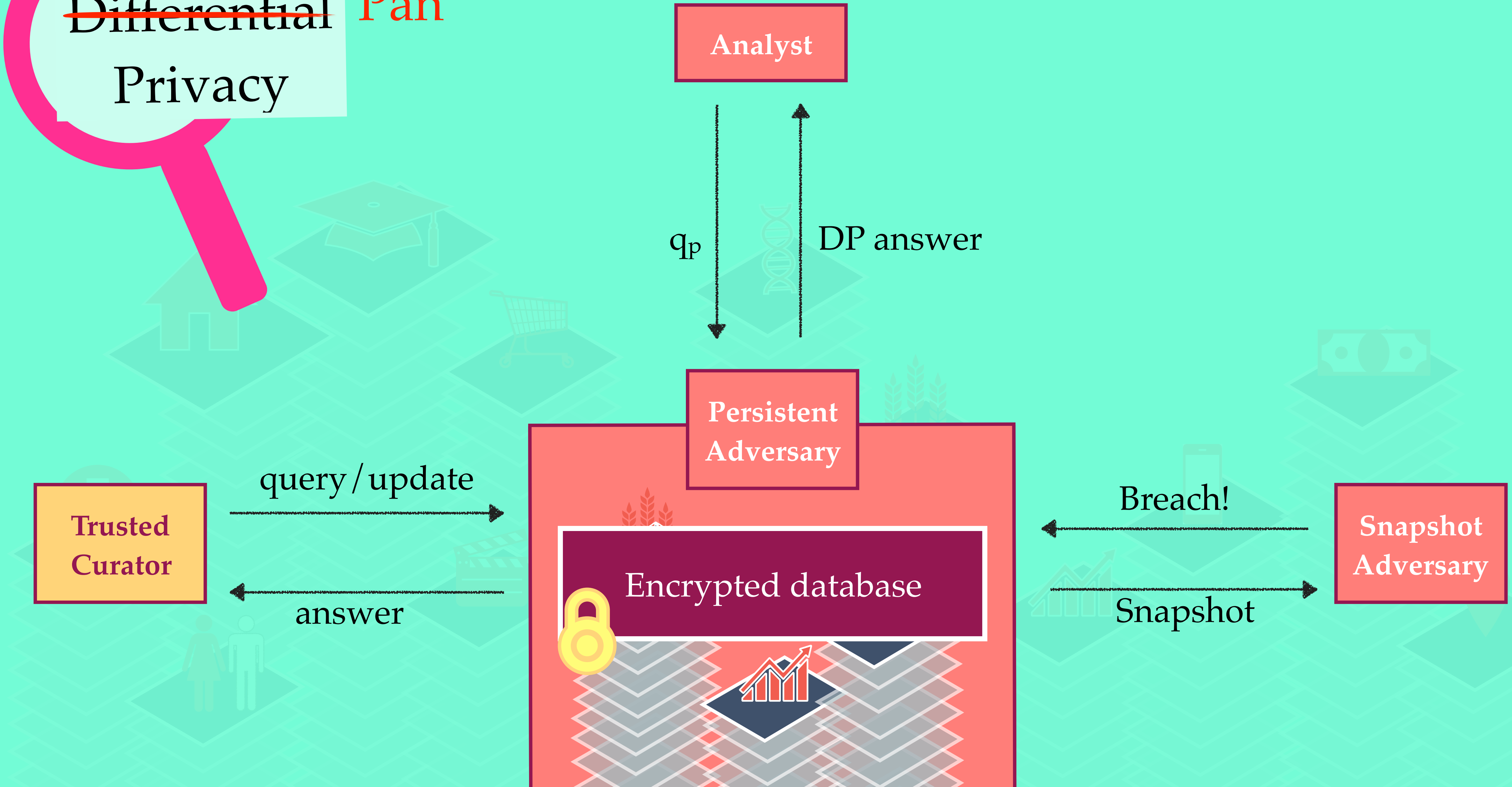
Pan-Private database

Breach!

Snapshot

Snapshot Adversary

~~Differential~~ Pan Privacy



Analyst

q_P

DP answer

Persistent Adversary

Trusted Curator

query/update

answer

Encrypted database

Breach!

Snapshot

Snapshot Adversary

Q: Can we design encrypted databases that support DP statistical queries?

STE schemes

Q: Can we design encrypted databases that support DP statistical queries?

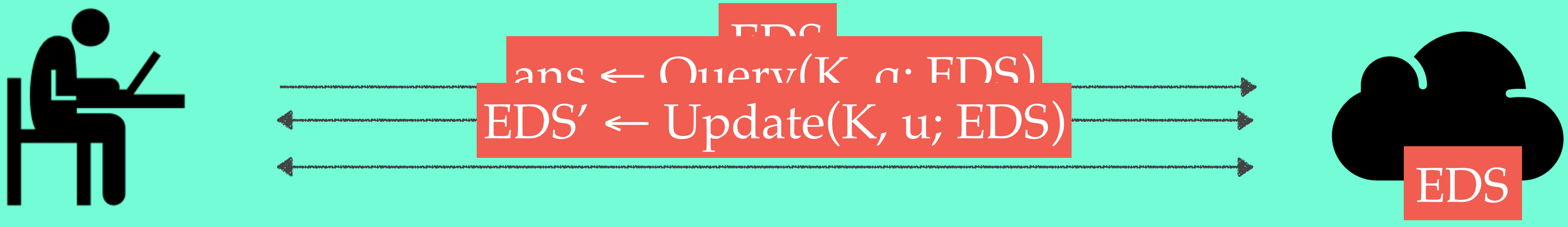
PSTE schemes

Outline

- STE Scheme
- PSTE scheme
    - Syntax + Security
- PSTE scheme for histogram queries (HPX)
    - Encrypted Private Counters (CPX)
- Efficiency Estimates

Structured Encryption [CK10]

STE = (Setup, Query, Update)



EDS

ans ← Query(K, q; EDS)

EDS′ ← Update(K, u; EDS)

EDS

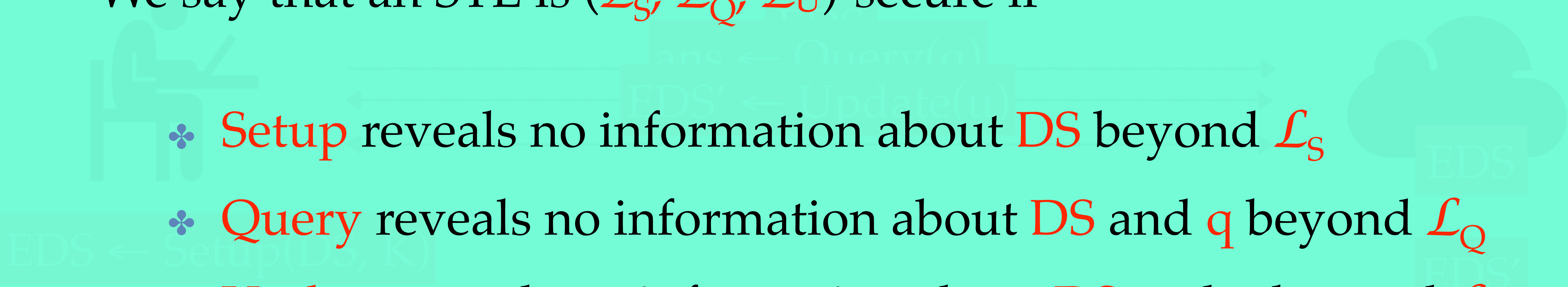
(EDS, K) ← Setup(DS)

EDS′

Structured Encryption [CK10]

STE = (Setup, Query, Update)

We say that an STE is ($\mathcal{L}_S$, $\mathcal{L}_Q$, $\mathcal{L}_U$)-secure if



- Setup reveals no information about DS beyond $\mathcal{L}_S$
- Query reveals no information about DS and q beyond $\mathcal{L}_Q$
- Update reveals no information about DS and u beyond $\mathcal{L}_U$

Outline

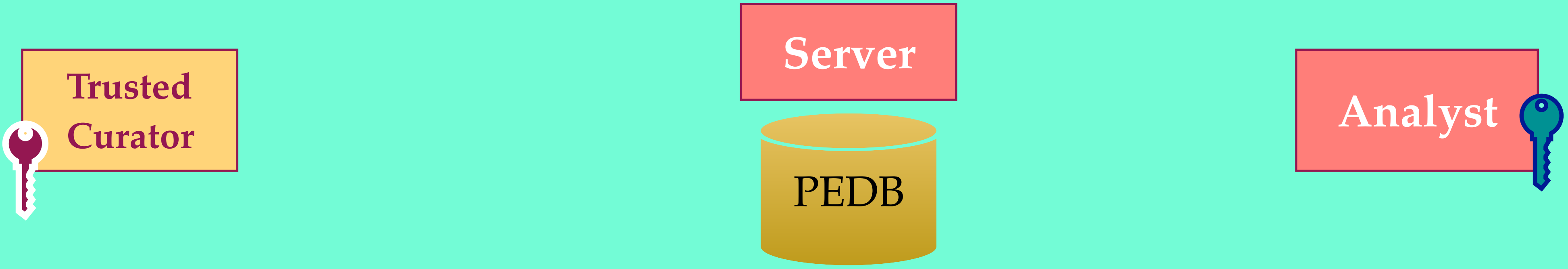
- STE Scheme
- PSTE scheme
  - Syntax + Security
- PSTE scheme for histogram queries (HPX)
  - Encrypted Private Counters (CPX)
- Efficiency Estimates

Private Structured Encryption

PSTE = (Setup, EAdd, ERemove, Equery, Pquery)

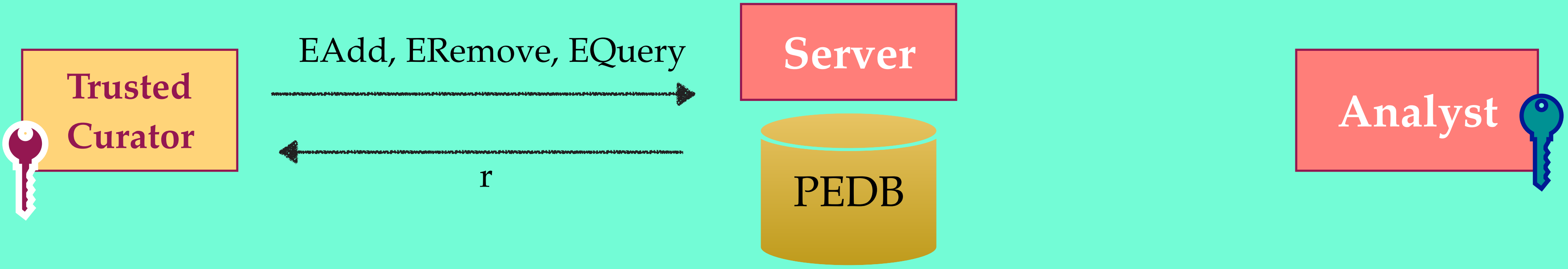
Private Structured Encryption

PSTE = (Setup, EAdd, ERemove, EQuery, PQuery)



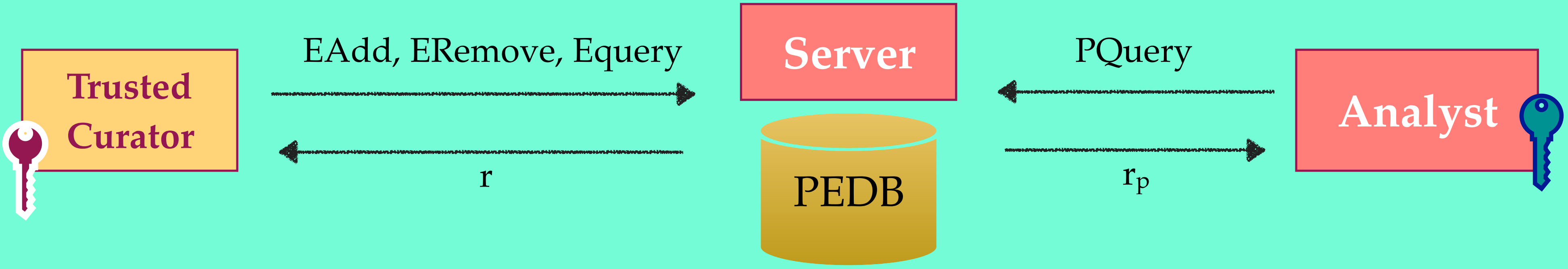
Private Structured Encryption

PSTE = (Setup, EAdd, ERemove, EQuery, PQuery)



Private Structured Encryption

PSTE = (Setup, EAdd, ERemove, EQuery, PQuery)



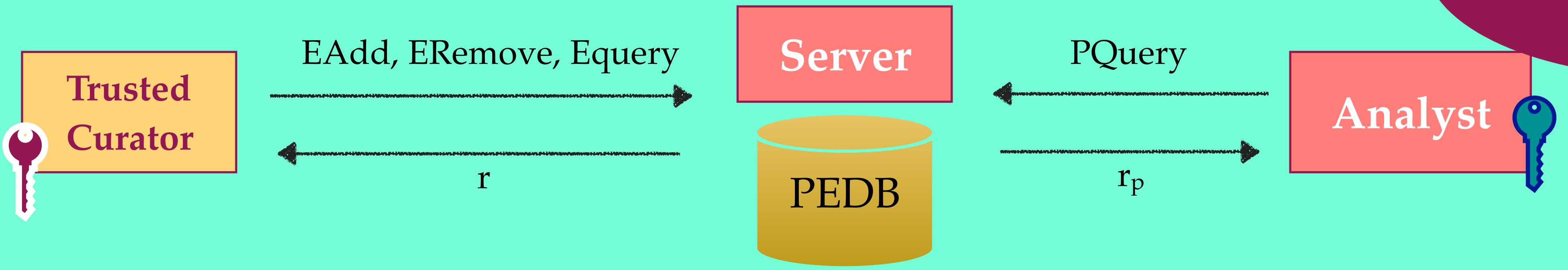
Private Structured Encryption: Correctness

PSTE = (Setup, EAdd, ERemove, EQuery, PQuery)

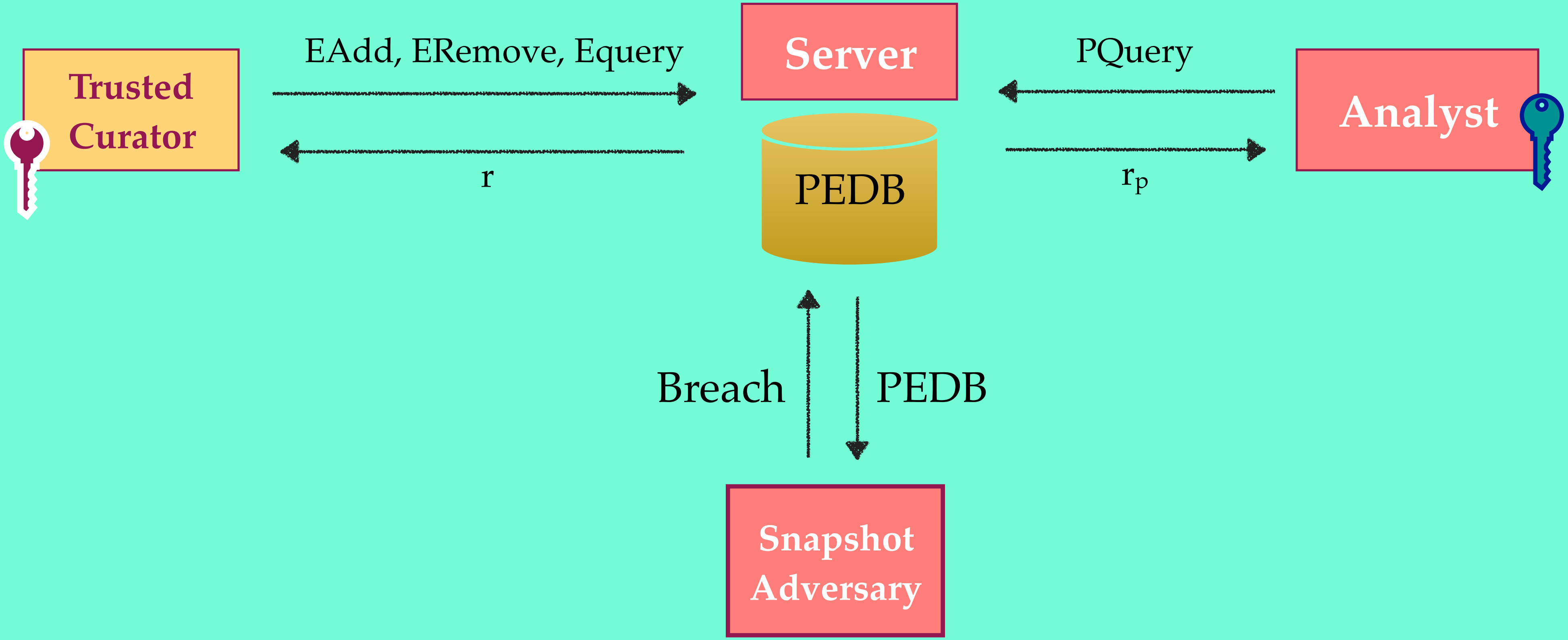
Should return the correct response

(α,δ)-useful

with prob. $\geq 1 - \delta$, $|r_p - r_a| \leq \alpha$



Private Structured Encryption: Security



Trusted Curator

EAdd, ERemove, Equery

r

Server

PEDB

PQuery

r_p

Analyst

Breach

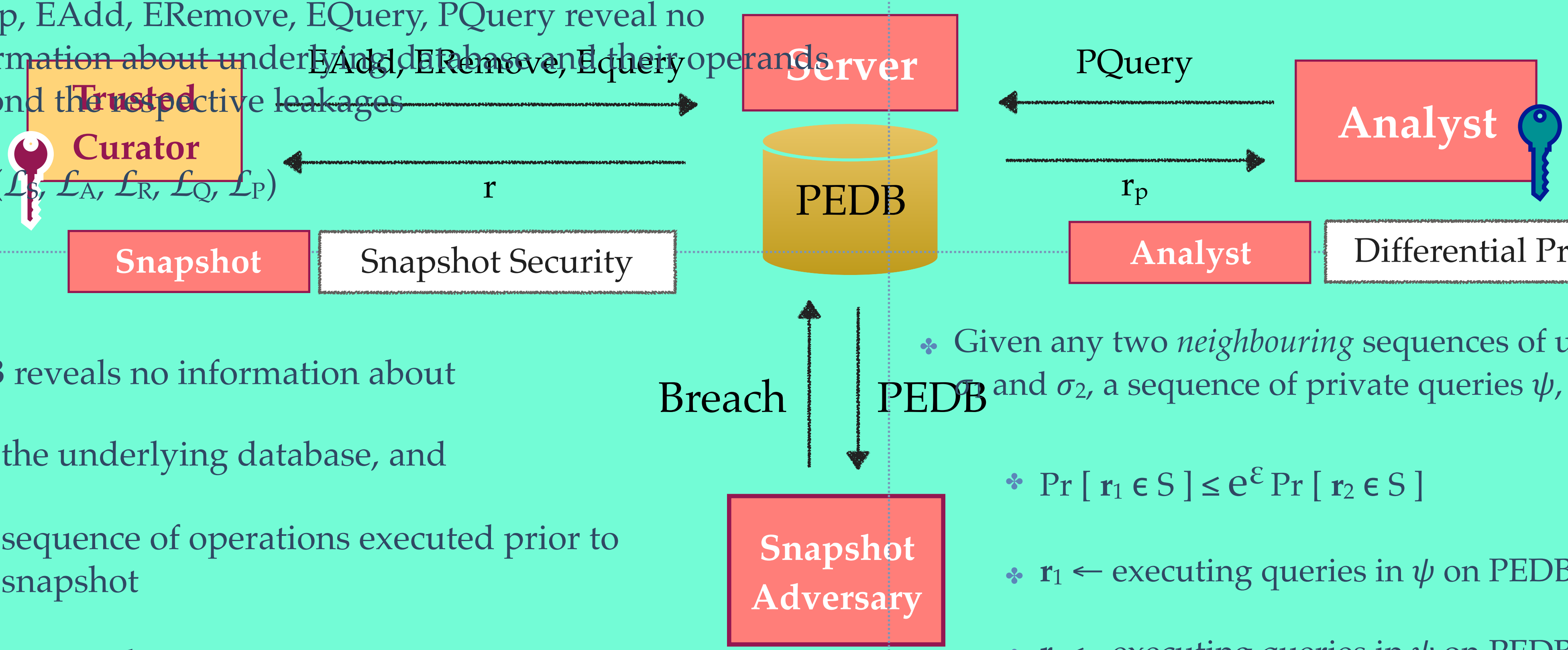
PEDB

Snapshot Adversary

Private Structured Encryption: Security

Server | Persistent Security

- Extension of STE security definitions
- Setup, EAdd, ERemove, EQuery, PQuery reveal no information about underlying database and their operands beyond the respective leakages
- $\mathcal{L} = (\mathcal{L}_S, \mathcal{L}_A, \mathcal{L}_R, \mathcal{L}_Q, \mathcal{L}_P)$



Snapshot | Snapshot Security

- PEDB reveals no information about
  - the underlying database, and
  - sequence of operations executed prior to snapshot
- beyond snapshot leakage $\mathcal{L}_{SN}$

Analyst | Differential Privacy

- Given any two *neighbouring* sequences of update operations σ_1 and σ_2, a sequence of private queries ψ, and a set S
  - Pr [$\mathbf{r}_1 \in$ S] $\leq e^{\varepsilon}$ Pr [$\mathbf{r}_2 \in$ S]
  - $\mathbf{r}_1 \leftarrow$ executing queries in ψ on PEDB resulting from σ_1
  - $\mathbf{r}_2 \leftarrow$ executing queries in ψ on PEDB resulting from σ_2

Outline

- STE Scheme
- PSTE scheme
  - Syntax + Security
- PSTE scheme for histogram queries (HPX)
  - Encrypted Private Counters (CPX)
- Efficiency Estimates

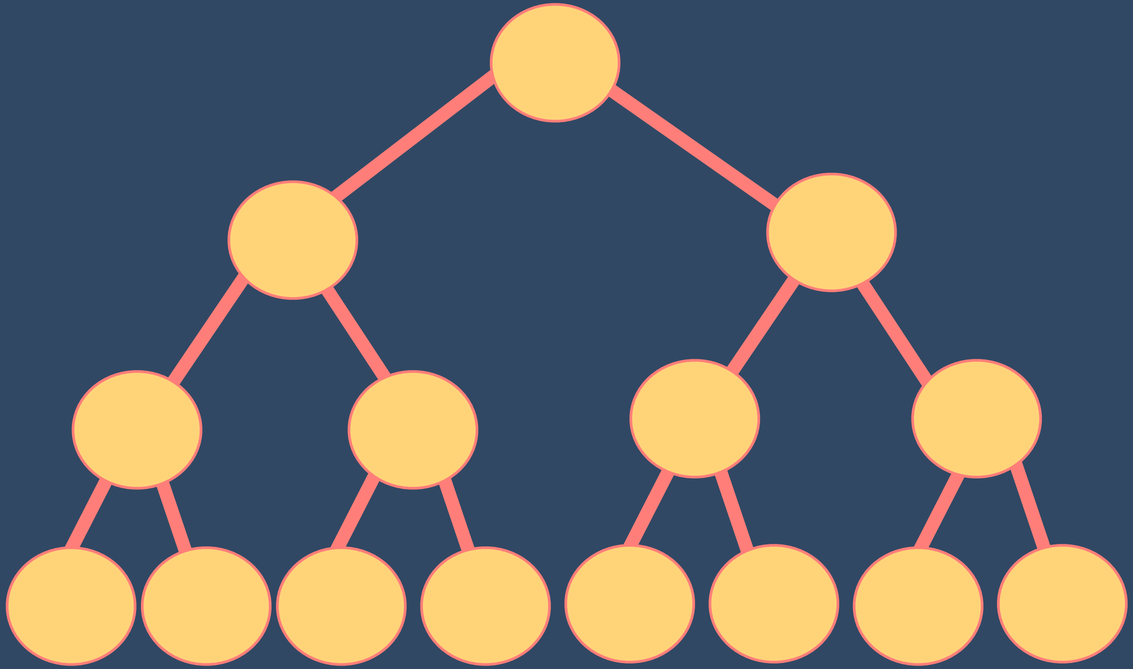
CPX: Encrypted Private Counter

Encrypted Private Counter (CPX)

- CPX = (Setup, EAdd, PRead)
- Encrypted DP counter
- EAdd(a) : a ϵ {1, -1, 0}
    - 1 : increment , -1 : decrement , 0 : no-op
- PRead reads the counter value which is DP
- Uses Binary Mechanism of [CSS' 11]

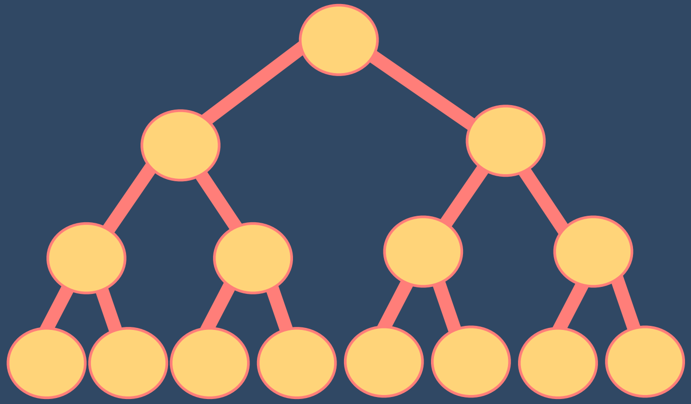
Binary Mechanism [CSS'11]

- Implements a private counter
- Add(a): a ϵ {1, -1, 0}
- PRead():
  - outputs DP counter value

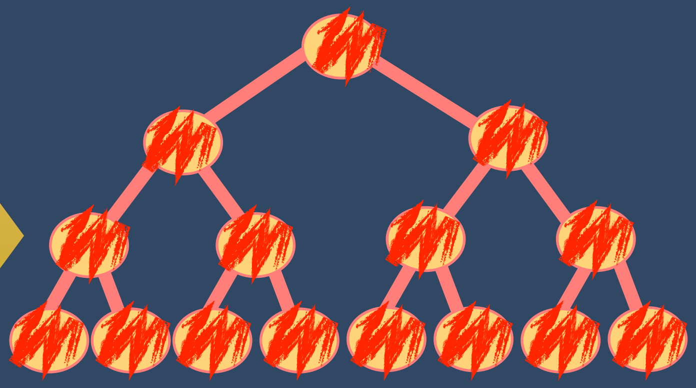


Private Counter

Enc Private Counter



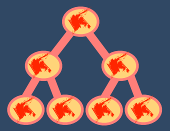
- Use AHE to encrypt each node
- EAdd(a): add ‘a’ homomorphically



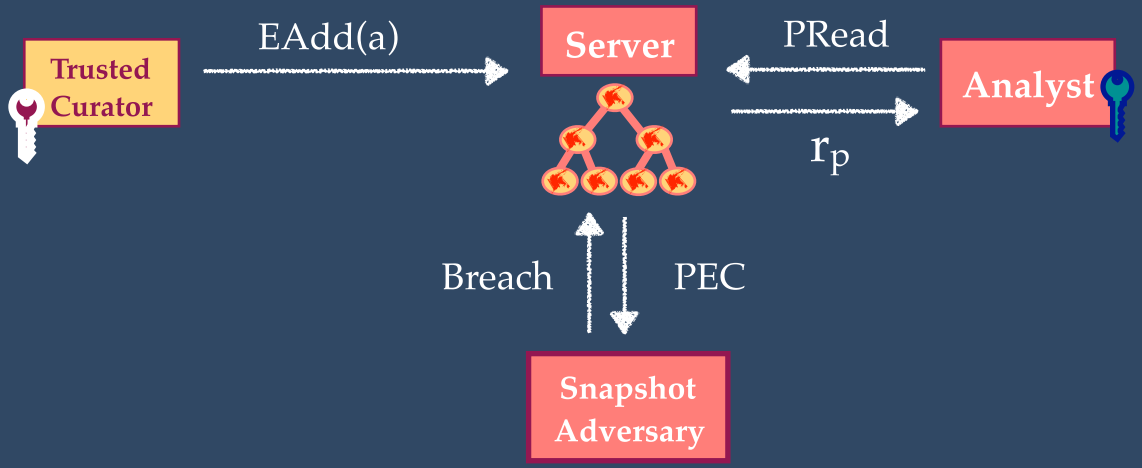
Encrypted Private Counter : Security

Server Persistent Security

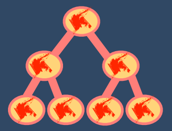
Enc(a)



Just learns that an add happened!



Snapshot Snapshot Security



Learns nothing !

Analyst Differential Privacy

Follows from DP of private counter!

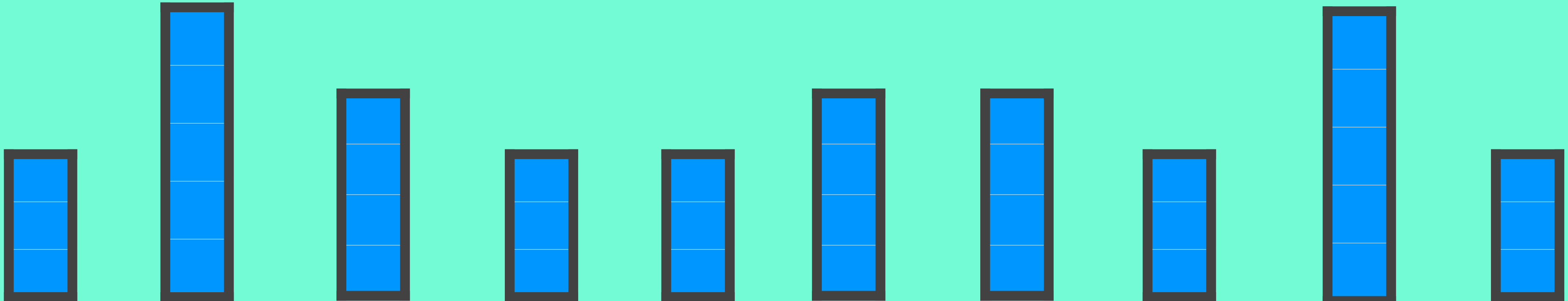
Outline

- STE Scheme
- PSTE scheme
    - Syntax + Security
- PSTE scheme for histogram queries (HPX)
    - Encrypted Private Counters (CPX)
- Efficiency Estimates

HPX: Encrypted Database for Private Histogram Queries

HPX: Encrypted Database for Private Histogram Queries

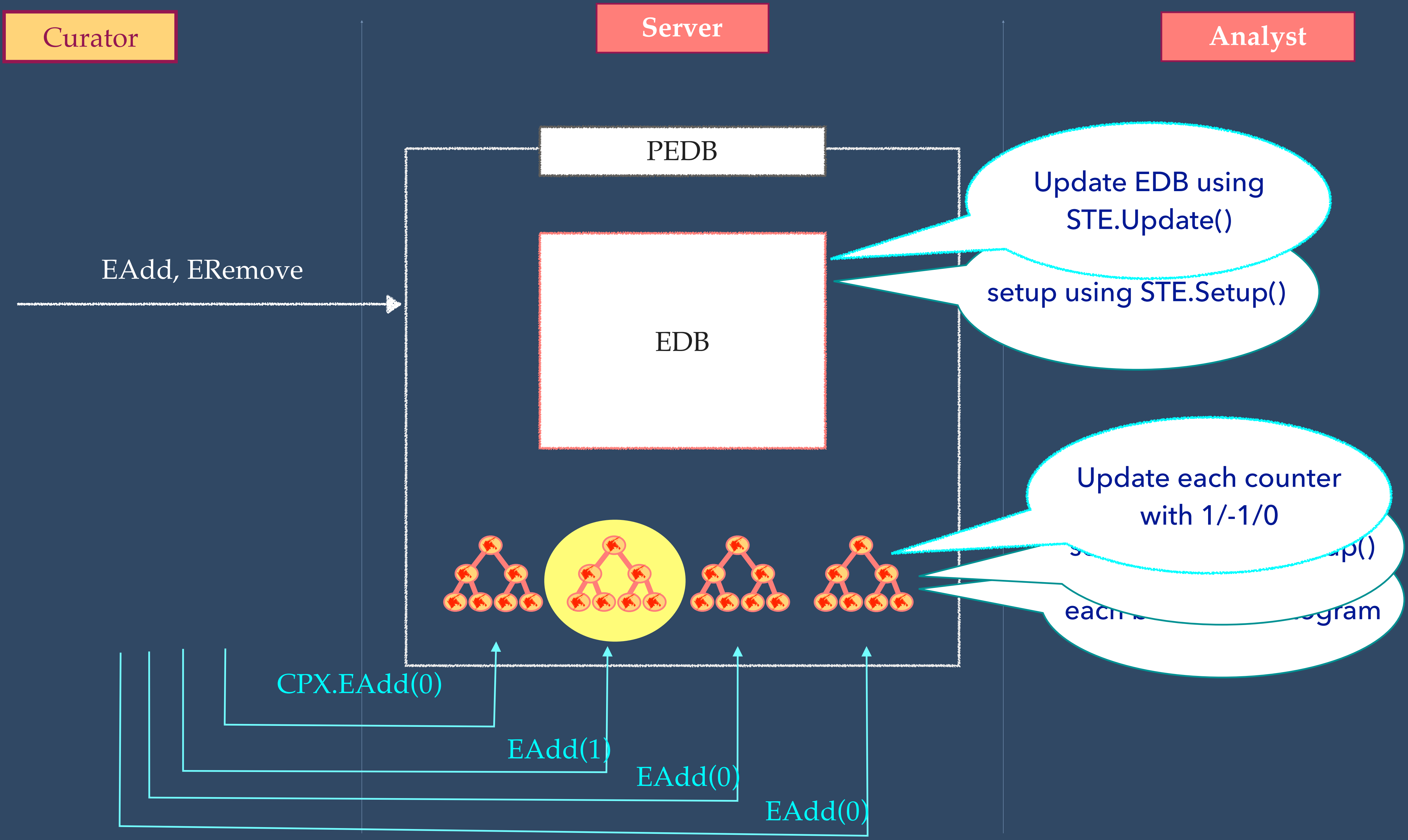
Count the number of elements satisfying a certain property



Dynamic STE$_{DB}$ scheme

+

Private Encrypted Counters (CPX)



Curator

Server

Analyst

PEDB

EDB

EAdd, ERemove

Update EDB using STE.Update()

setup using STE.Setup()

Update each counter with 1/-1/0

s p()

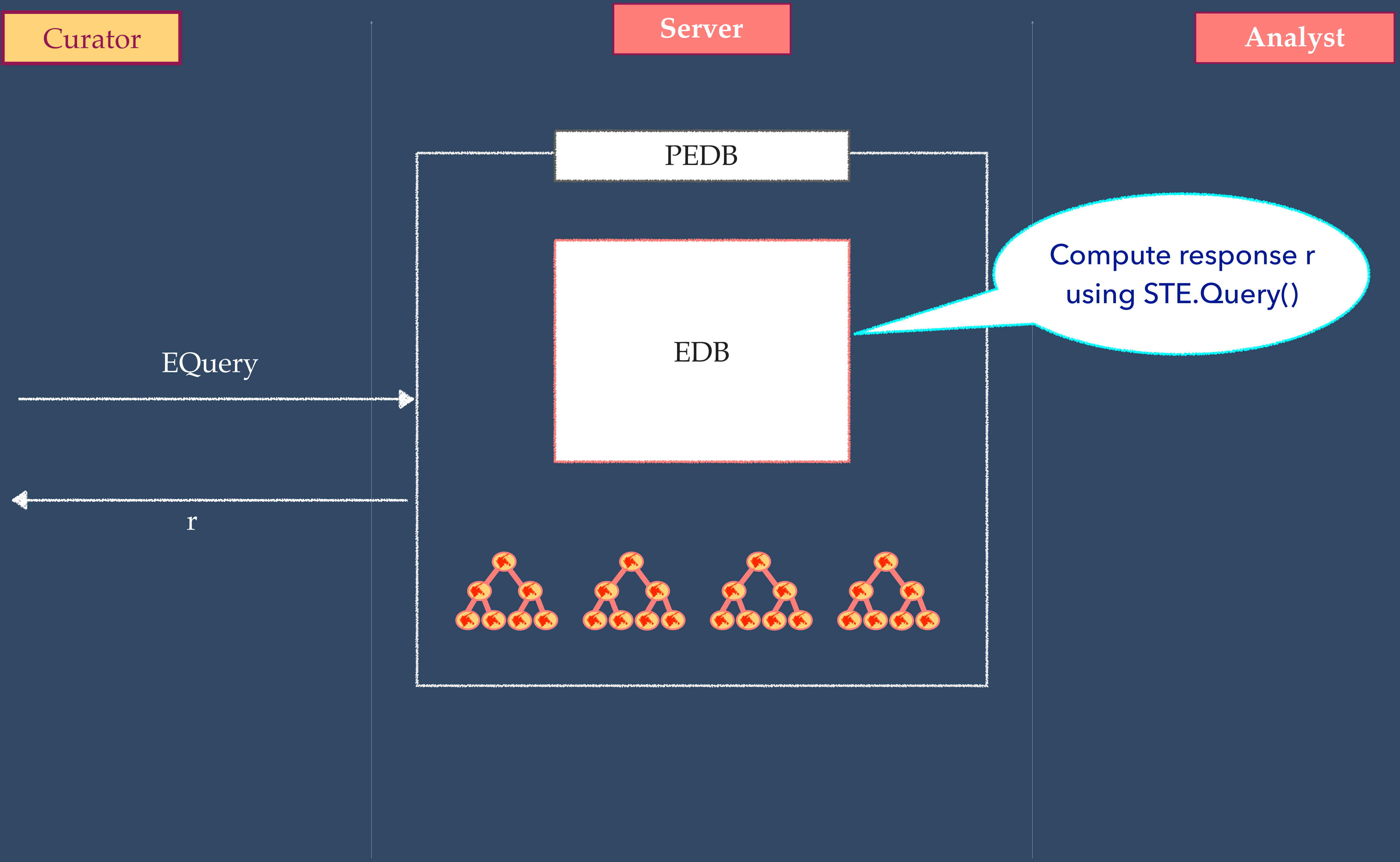
each b gram

CPX.EAdd(0)

EAdd(1)

EAdd(0)

EAdd(0)



Curator

Server

Analyst

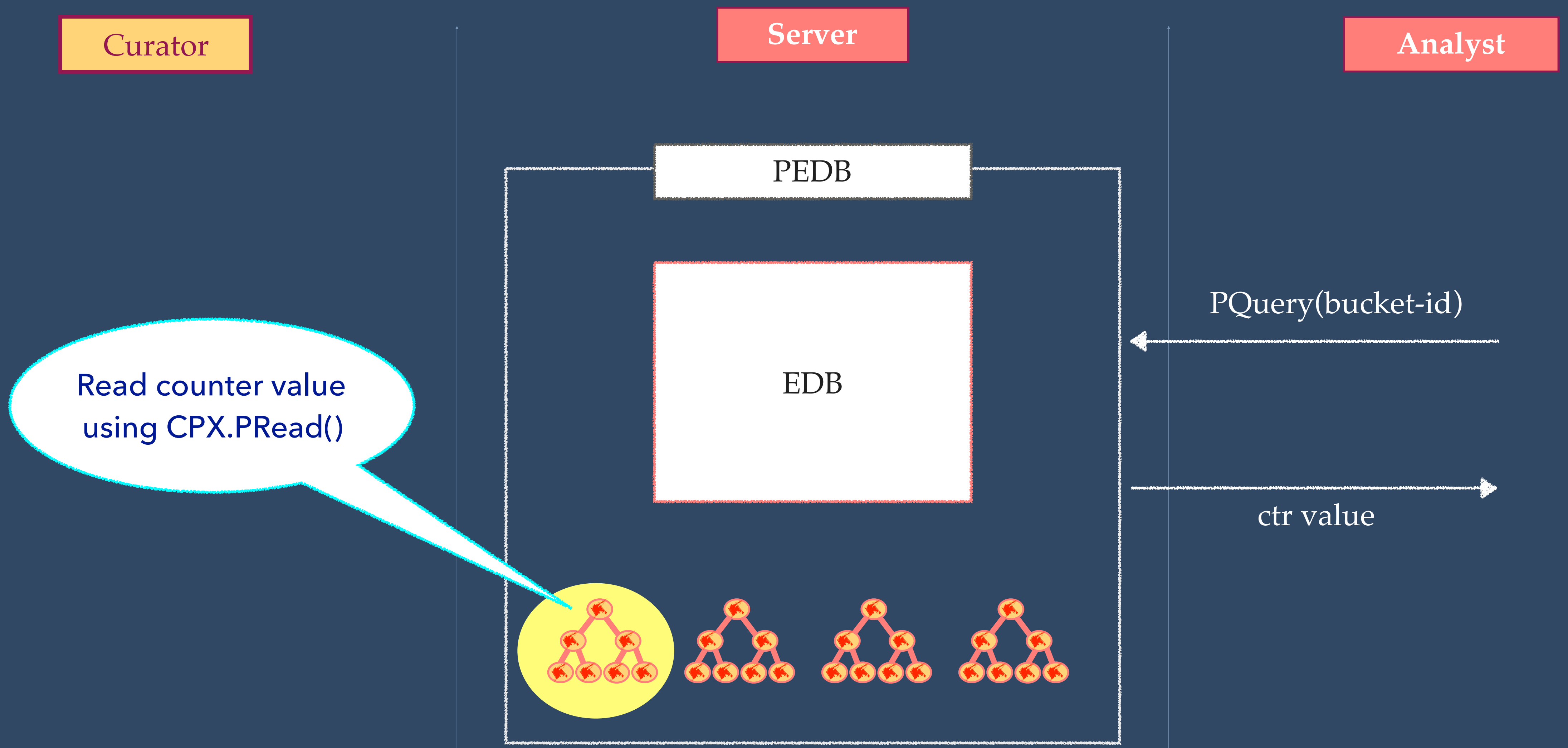
PEDB

EDB

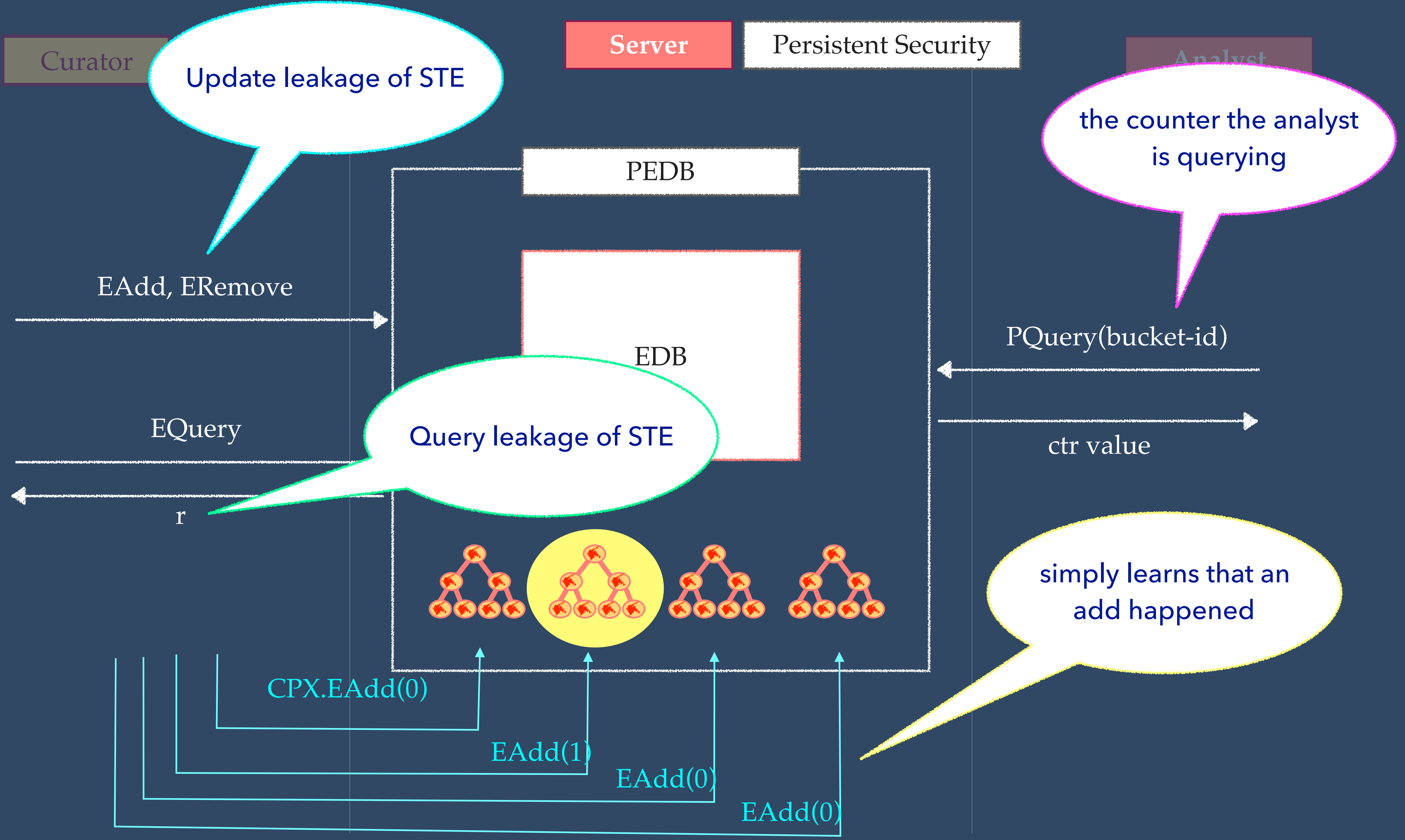
Compute response r using STE.Query()

EQuery

r


Curator
Server
Analyst
PEDB
EDB
PQuery(bucket-id)
ctr value
Read counter value using CPX.PRead()

HPX: Security



Server

Persistent Security

Curator

Update leakage of STE

Analyst

the counter the analyst is querying

PEDB

EAdd, ERemove

EDB

PQuery(bucket-id)

EQuery

Query leakage of STE

ctr value

r

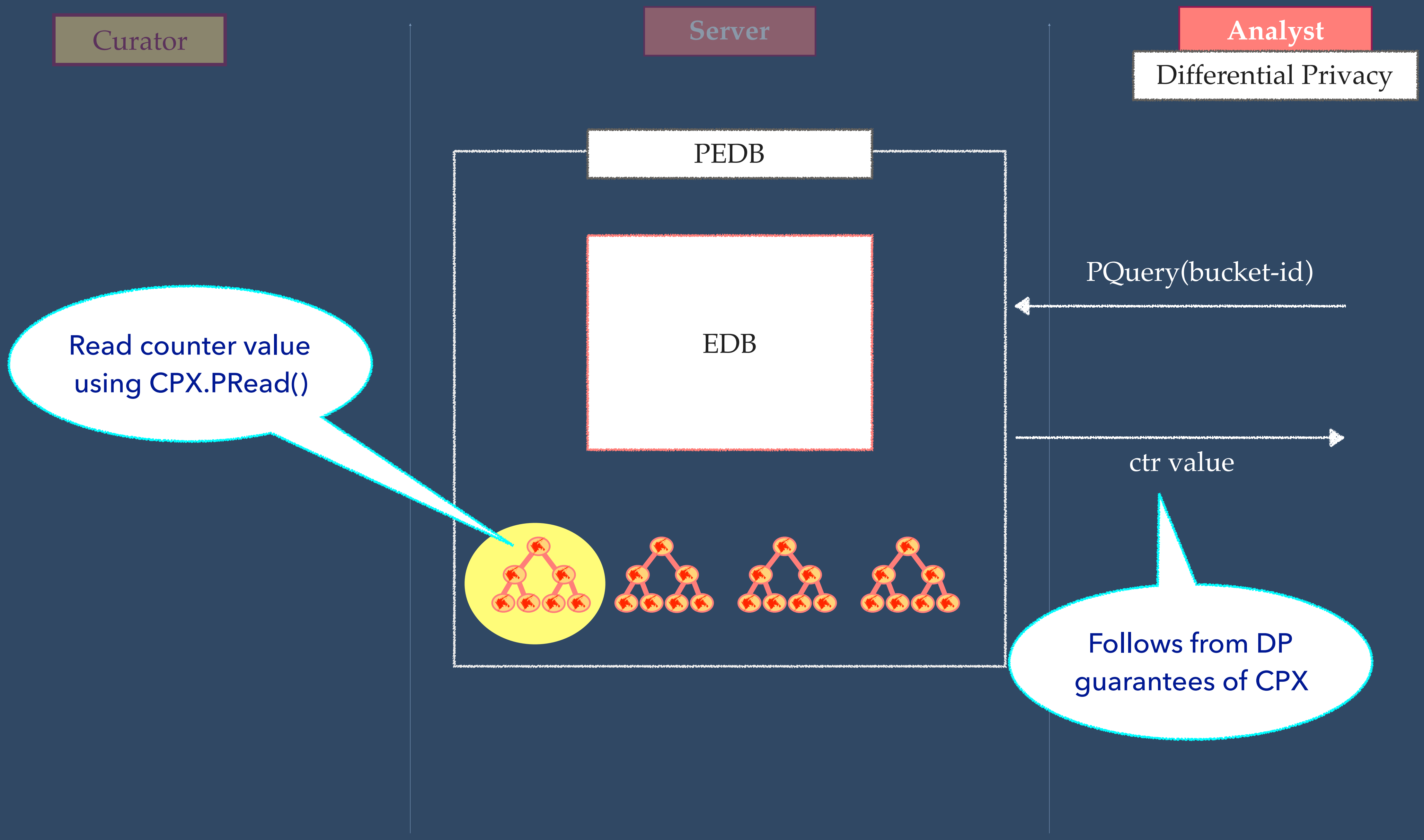
simply learns that an add happened

CPX.EAdd(0)

EAdd(1)

EAdd(0)

EAdd(0)



Curator

Server

Analyst

Differential Privacy

PEDB

EDB

PQuery(bucket-id)

ctr value

Read counter value using CPX.PRead()

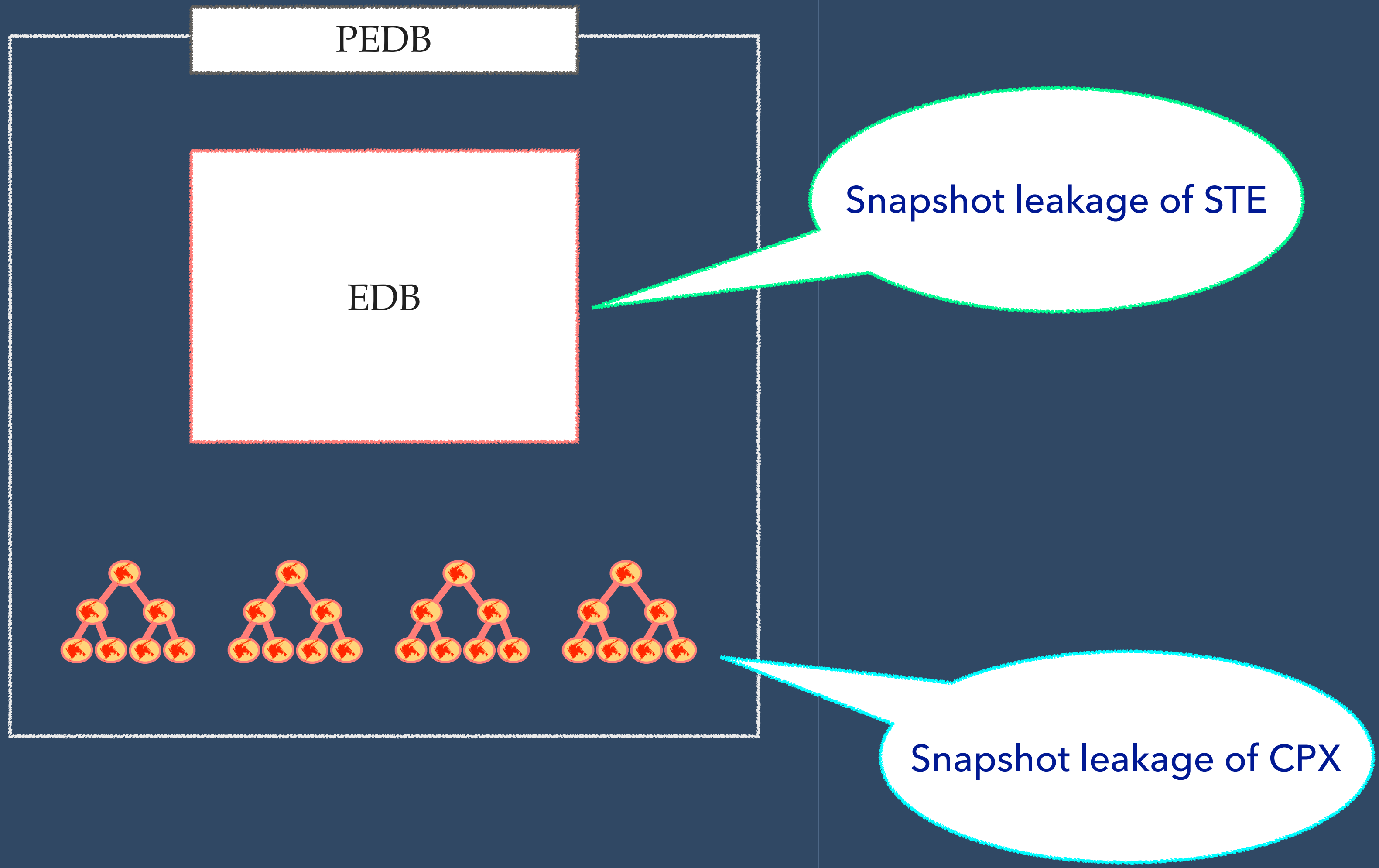
Follows from DP guarantees of CPX

Snapshot

Snapshot Security

Server

Analyst



PEDB

EDB

Snapshot leakage of STE

Snapshot leakage of CPX

Outline

- STE Scheme
- PSTE scheme
    - Syntax + Security
- PSTE scheme for histogram queries (HPX)
    - Encrypted Private Counters (CPX)
- Efficiency Estimates

Efficiency Estimates

$$\mathrm{time}^{\mathrm{add}}_{\mathrm{HPX}}(\mathrm{v}) = \mathrm{time}^{\mathrm{add}}_{\mathrm{DB}}(\mathrm{v}) + \mathrm{n} \cdot \mathrm{time}^{\mathrm{add}}_{\mathrm{ctr}}(\lambda)$$

0.945 ms

$$\mathrm{time}^{\mathrm{rem}}_{\mathrm{HPX}}(\mathrm{v}) = \mathrm{time}^{\mathrm{rem}}_{\mathrm{DB}}(\mathrm{v}) + \mathrm{n} \cdot \mathrm{time}^{\mathrm{add}}_{\mathrm{ctr}}(\lambda)$$

0.945 ms

$$\mathrm{time}^{\mathrm{qry}}_{\mathrm{HPX}}(\mathrm{q}) = \mathrm{time}^{\mathrm{qry}}_{\mathrm{DB}}(\mathrm{q})$$

1 microsecond

$$\mathrm{time}^{\mathrm{pqry}}_{\mathrm{HPX}}(\mathrm{pq}) = \mathrm{time}^{\mathrm{pread}}_{\mathrm{ctr}}(\lambda)$$

21.17 ms

- AHE : Paillier with 2048-bit key
- STE scheme : DLS from [AKM '19] where DB = MM
- max operations : 2^{32}
- n = 25
- MM size : 10 million pairs

Collusions

- Snapshot + Analyst
- Persistent + Analyst
- Use stronger STE schemes

Thank You!!