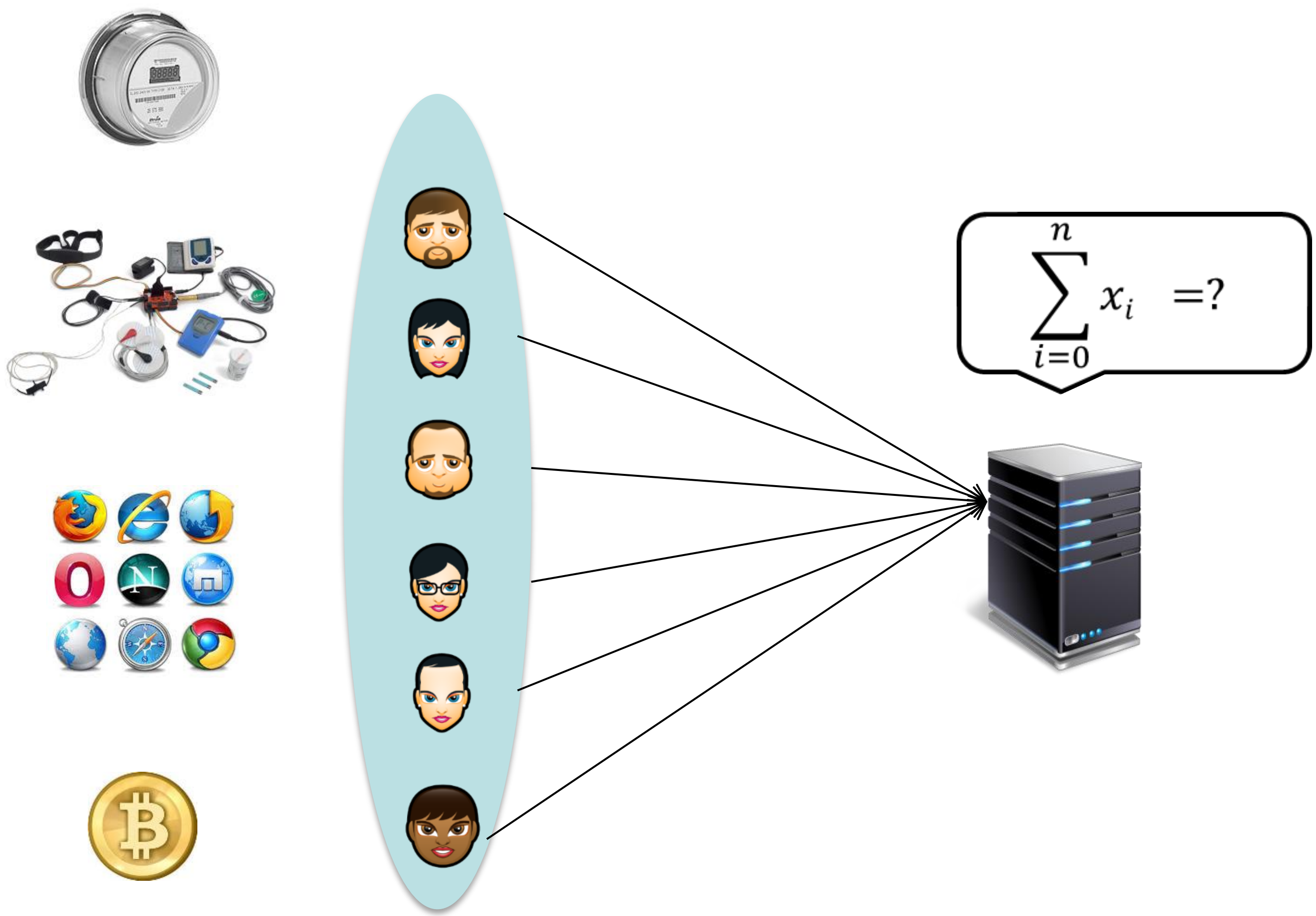
Private and Dynamic Time-Series Data Aggregation with Trust Relaxation

I. Leontiadis, K. Elkhiyaoui, R. Molva

Problem



Energy forecasting

Medical data analysis

User Profiling

Overall Transactions

EURECOM Sophia Antipolis

Existing solutions

- **Trusted Aggregator**
  - Unrealistic
- **Secret sharing [ET2012]**
  - Increased communication costs
- **Differential privacy [JK2012, CSS2012, RN2010]**
  - Orthogonal to our goal
- **Trusted Dealer [SCRCS2011, JL2013, GMP2014]**
  - Strong trust assumption

State of the Art [SCRCS2011]

- **Setup(k):**
  - $\mathbb{G}$ a cyclic group with a generator $\boldsymbol{g}$ and prime order p
  - Trusted Dealer distributes:
    - ☞secret keys $\mathrm{sk_i} \in \mathbb{Z}_p$.
    - ☞$sk_0 = -\sum_{i=1}^{n} sk_i$ to the Aggregator.
    - ☞$H(\quad): \{0,1\}^* \rightarrow \mathbb{G}$

- **Encrypt($\boldsymbol{x_i}, \boldsymbol{t}$):**
  - $c_{i,t} = g^{x_i} H(t)^{sk_i} \in \mathbb{Z}_p$

- **Aggregate:**
  - $V = H(t)^{sk_0} \prod_{i=1}^{n} c_{i,t} = g^{\sum_{i=1}^{n} c_{i,t}} g \in \mathbb{Z}_p$
  - $\sum_{i=1}^{n} c_i = \log_g(V)$

- Vulnerable to user failures
- No dynamicity
- Expensive decryption
- Fully trusted dealer

EURECOM Sophia Antipolis

State of the Art contd. [JL2013]

- **Setup(k):**
  - $\mathbf{N} = \mathbf{pq}$ for primes $\mathbf{p}, \boldsymbol{q}$ (l the size of $\mathbf{N}$)
  - Trusted Dealer distributes:
    - secret keys $\mathrm{sk_i} \in \{0,1\}^{2l}$ to the users.
    - $sk_0 = -\sum_{i=1}^{n} sk_i$ to the Aggregator.
    - $H(\quad): \mathbb{Z}_N \rightarrow (\mathbb{Z}_{N^2})^*$
- **Encrypt($\boldsymbol{x_i}, \boldsymbol{t}$):**
  - $c_{i,t} = (1 + x_{i,t}N)H(t)^{sk_i} mod\ N^2$
- **Aggregate:**
  - $V_t = H(t)^{sk0} \prod_{i=1}^{n} c_{i,t} = (1 + \sum_{i=1}^{n} x_{i,t}\, N) mod\ N^2$
  - $\sum_{i=1}^{n} x_{i,t} = \frac{V_t - 1}{N} \in \mathbb{Z}$

- Vulnerable to user failures
- No dynamicity
- Fully trusted dealer

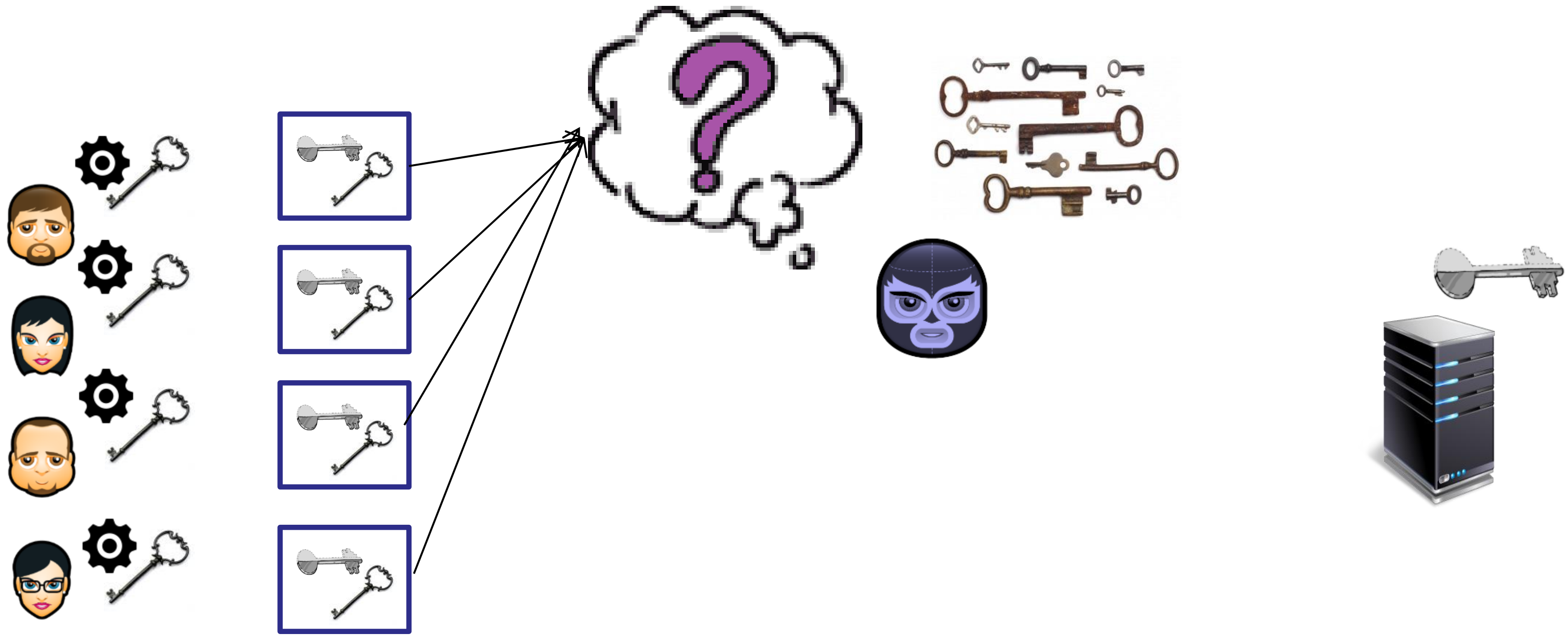
Drawbacks of previous solutions

- **Functionality**
  - No Dynamic group management
  - Not resilient to user failures

- **Privacy**
  - Fully trusted key dealer

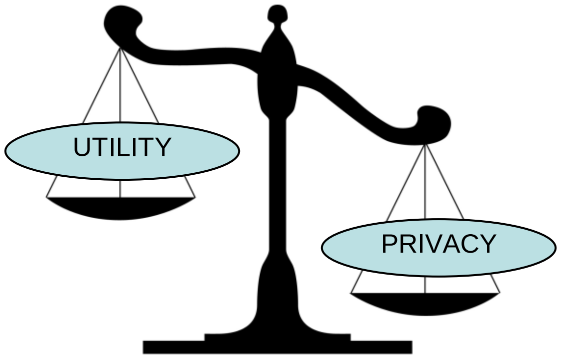
Idea of Solution

- **Users generate their secret keys.**
- **Semi trusted Collector.**
- **Blinded secret keys.**



Privacy Requirements

- **Aggregator obliviousness:**
  - Aggregator learns nothing but the aggregate.
- **Collector obliviousness:**
  - Collector learns nothing.



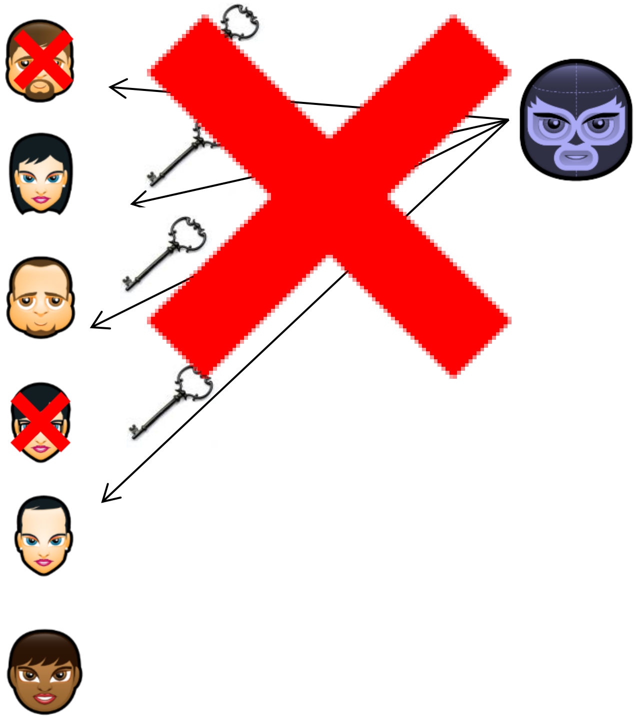
EURECOM
Sophia Antipolis

Functionality Enhancements

- **Dynamicity**

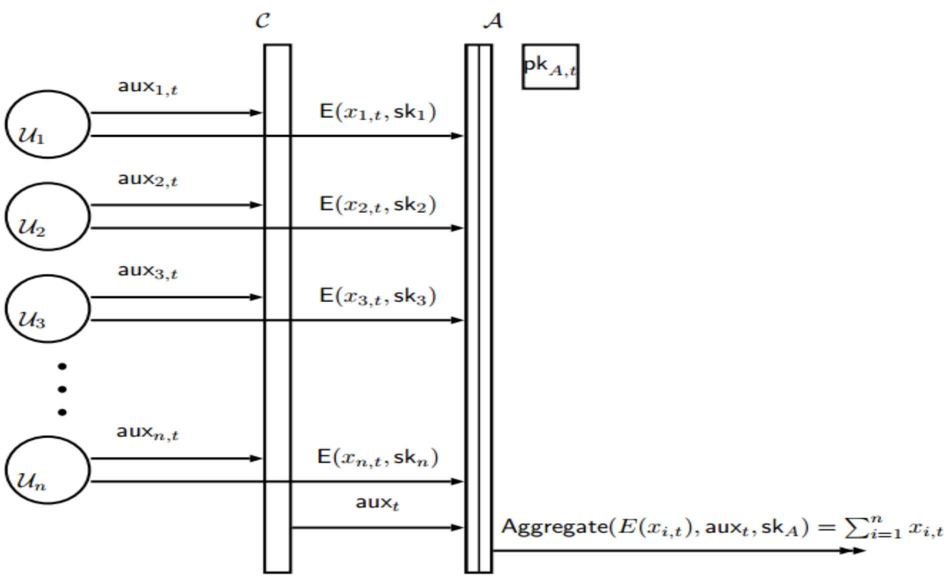
Functionality Enhancements

- **Dynamicity**
- **Fault-Tolerance**



Tools

1. **JL encryption.**
2. **Self-generated keys (by users).**
3. **Responsibility splitting mechanism.**

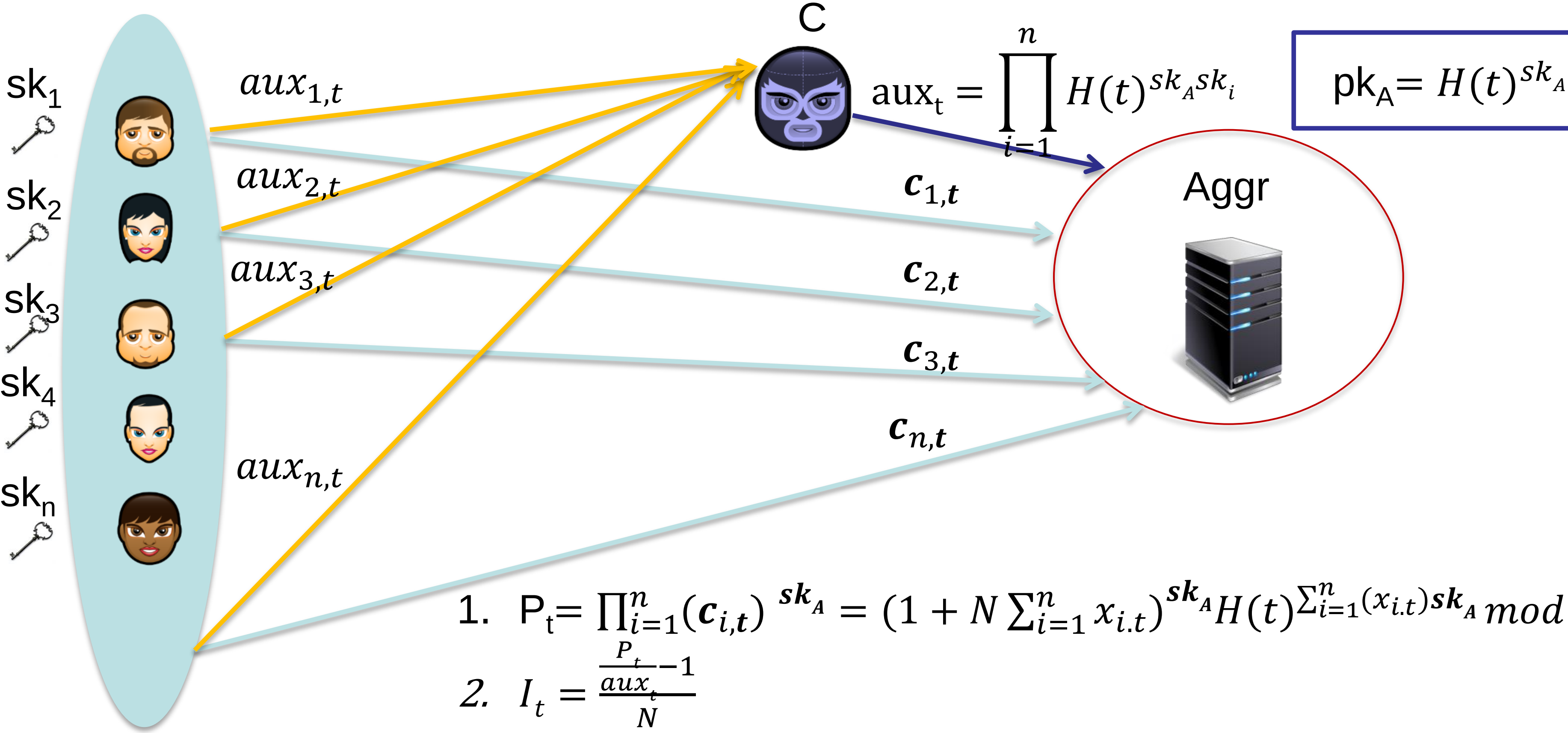


Our scheme

$$aux_{i,t} = H(t)^{sk_A sk_i}$$

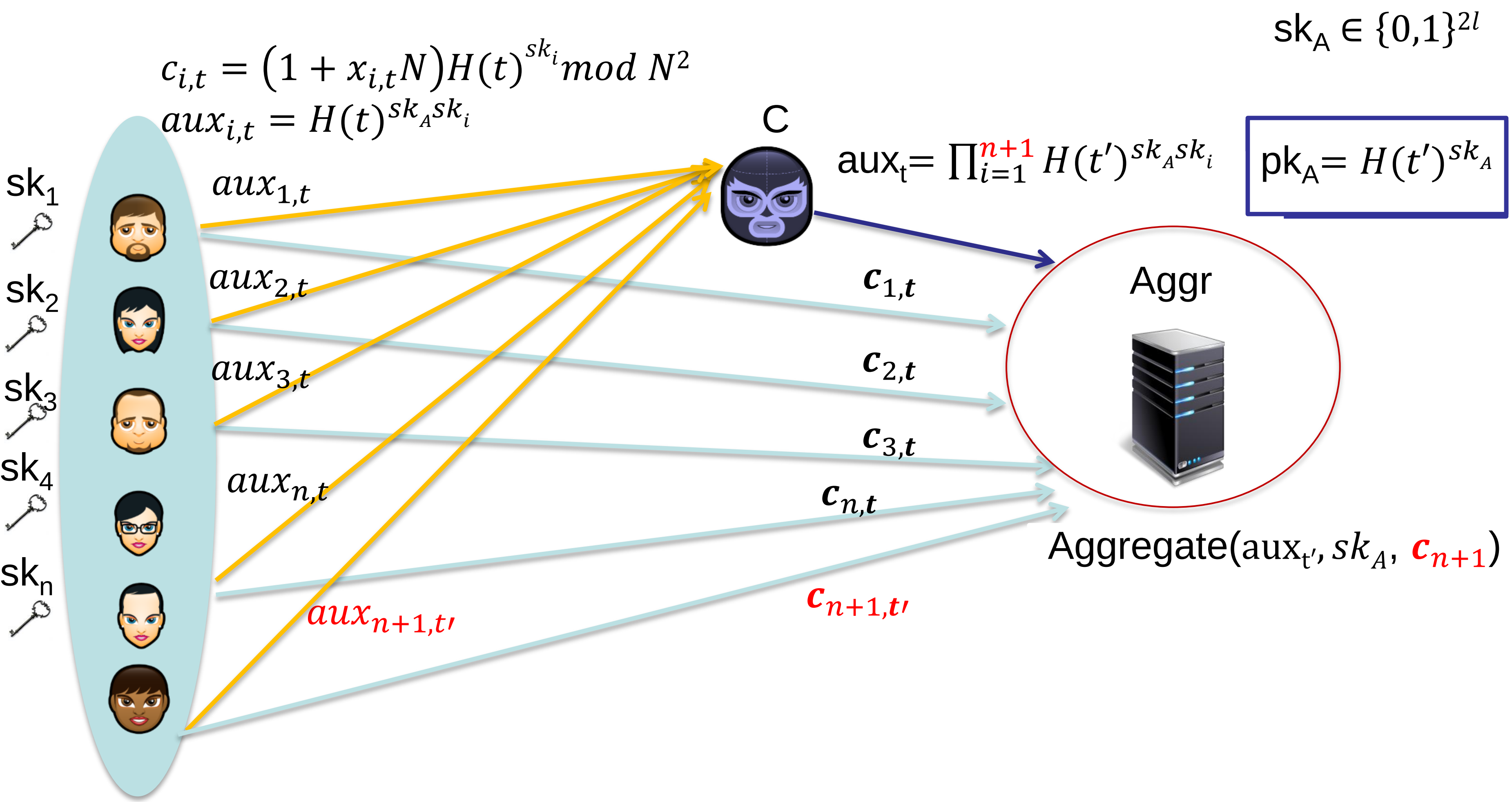
$$c_{i,t} = (1 + x_{i,t}N)H(t)^{sk_i} mod\ N^2$$

$sk_A \in \{0,1\}^{2l}$



1. $P_t = \prod_{i=1}^{n}(\boldsymbol{c_{i,t}})^{\boldsymbol{sk_A}} = (1 + N\sum_{i=1}^{n} x_{i.t})^{\boldsymbol{sk_A}} H(t)^{\sum_{i=1}^{n}(x_{i.t})\boldsymbol{sk_A}} mod\ N^2$
2. $I_t = \frac{\frac{P_t}{aux_t} - 1}{N}$
3. $\sum_{i=1}^{n} x_{i.t} = I_t \boldsymbol{sk_A}^{-1}\ mod\ \mathbb{Z}_N$

Dynamicity and Resiliency to failures



Privacy analysis

- **Aggregator Obliviousness based on:**
  - DCR in $(\mathbb{Z}/\mathbb{Z}_{N^2})^*$,

- **Collector Obliviousness based on:**
  - DCR in $(\mathbb{Z}/\mathbb{Z}_{N^2})^*$,
  - QR in $\mathbb{Z}_N^*$
  - DDH in the subgroup of QR in $(\mathbb{Z}_N)^*$

Evaluation

- **Theoretical**

Entity	Computation	Communication
User	2 EXP $+1$ MULT $+1$ ADD $+1$HASH	$4 \cdot l$
Aggregator	2 EXP $+2$ DIV $+(n-1)$ MULT $+1$HASH	$2 \cdot l$
Collector	$(n-1)$ MULT	$2 \cdot l$

- **Experimental (Charm framework on Python 3.2.3, Intel Core i5 CPU M 560 @ 2.67GHz 4Cores with 8GB of memory running Ubuntu 12.04 32bit)**

N \ Values	[1-10]	[1-100]	[1-1000]
1024	**110.13** μs	**112.23** μs	**114.57** μs
2048	**116.50** μs	**117.15** μs	**118.34** μs
3072	**116.99** μs	**118.23** μs	**120.83** μs

Encryption time per user

N \ Users	350	700	1000	2500
1024	**0.26** s	**2.40** s	**9.65** s	**49.92** s
2048	**0.65** s	**5.82** s	**24.16** s	**123.19** s
3072	**1.01** s	**9.37** s	**39.34** s	**198.12** s

Aggregation time

Recap

- **Aggregation of time series data**
  - Fast
  - Dynamic
  - Resilient to user failures
  - Relaxed trust assumption
- **How?**
  - $(1+N)^x = 1 + Nx \ mod \ N^2$ [JL2013].
  - Users generate keys independently.
  - Responsibility splitting mechanism.
  - Untrusted collector.

Looking Ahead

- **Advanced aggregate functionalities**
- **Verifiability**
- **Collusion resistant (?)**

Questions?

Thank you!!!

Iraklis Leontiadis
leontiad@eurecom.fr

EURECOM
Sophia Antipolis

References

- **[GMP2014]: Privacy-Enhanced Participatory Sensing with Collusion Resistance and Data Aggregation** [CANS2014]
- **[ET2012]: Private Computation of Spatial and Temporal Power Consumption with Smart Meters.** [ACNS 2012]
- **[JK2012]: Fault-Tolerant Privacy-Preserving Statistics.** [Privacy Enhancing Technologies 2012]
- **[CSS2012]: Privacy-Preserving Stream Aggregation with Fault Tolerance.** [Financial Cryptography 2012]
- **[RN2010]: Differentially private aggregation of distributed time-series with transformation and encryption.** [SIGMOD Conference 2010]
- **[SCRCS2011]: Privacy-Preserving Aggregation of Time-Series Data.** [NDSS 2011]
- **[JL2013]: A Scalable Scheme for Privacy-Preserving Aggregation of Time-Series Data.** [Financial Cryptography 2013]