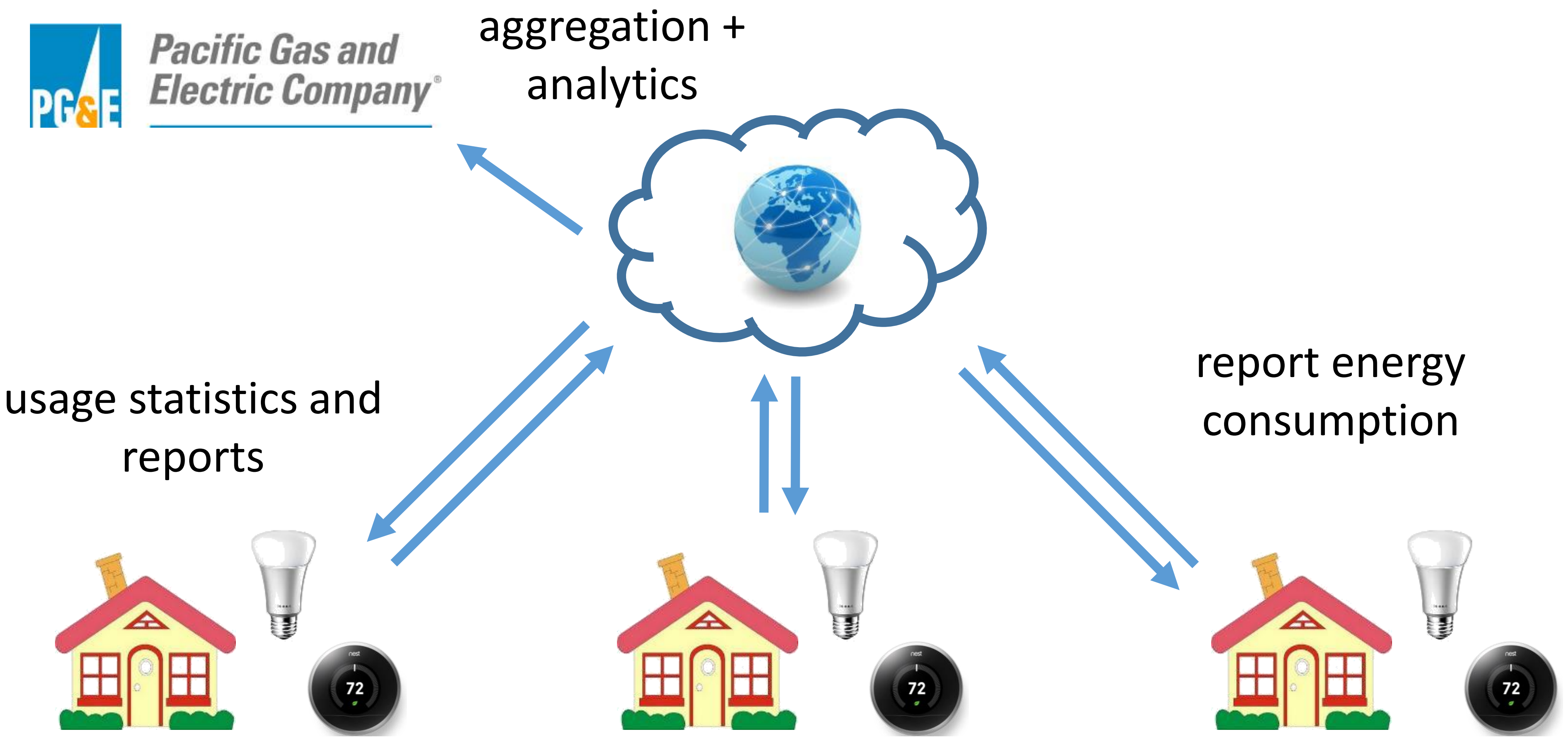
Computing on Encrypted Data

Secure Internet of Things Seminar

David Wu

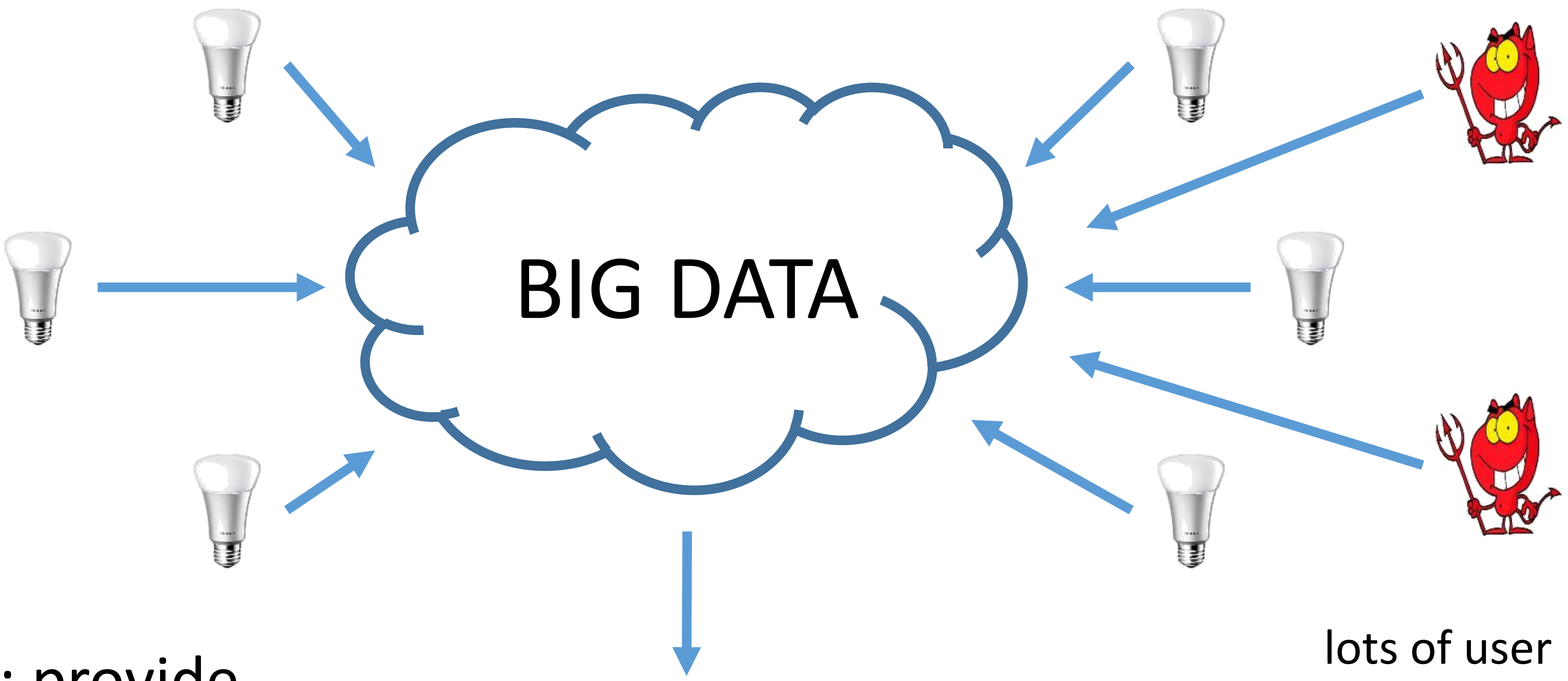
January, 2015

New Applications in the Internet of Things



Smart Homes

The Power of the Cloud



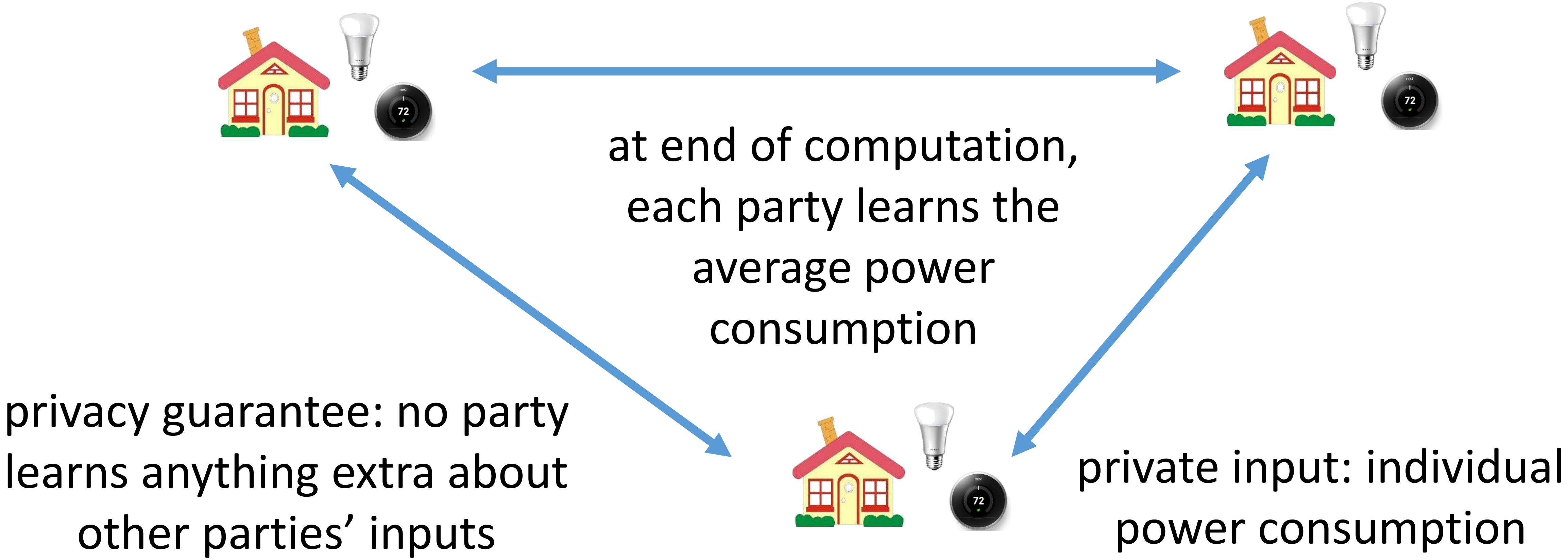
Question: provide service, preserve privacy

analytics
recommendations
personalization

lots of user information = big incentives

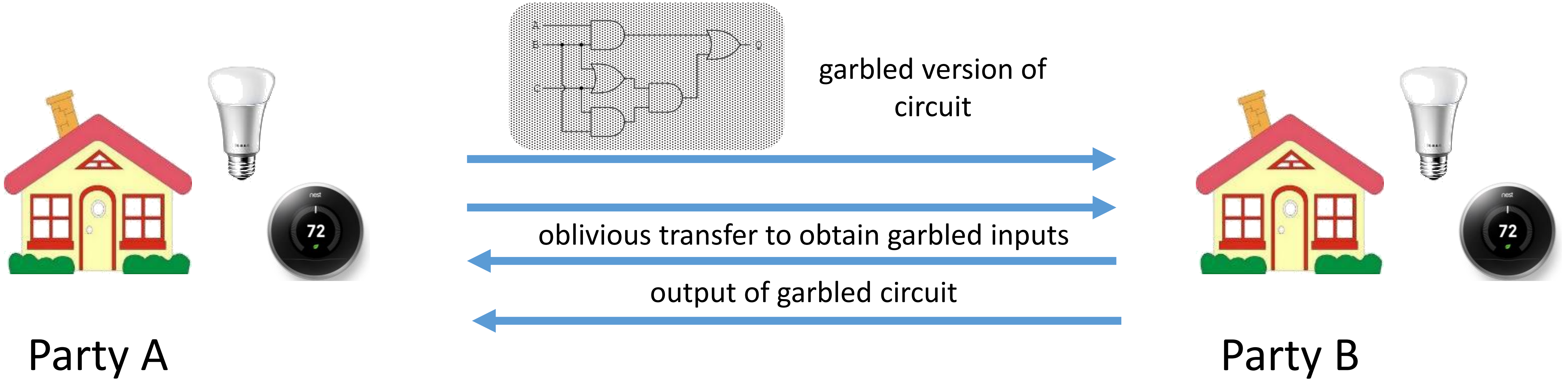
Secure Multiparty Computation (MPC)

Multiple parties want to compute a joint function on *private* inputs



Two Party Computation (2PC)

- Simpler scenario: two-party computation (2PC)
- 2PC: Mostly “solved” problem: Yao’s circuits [Yao82]
  - Express function as a Boolean circuit



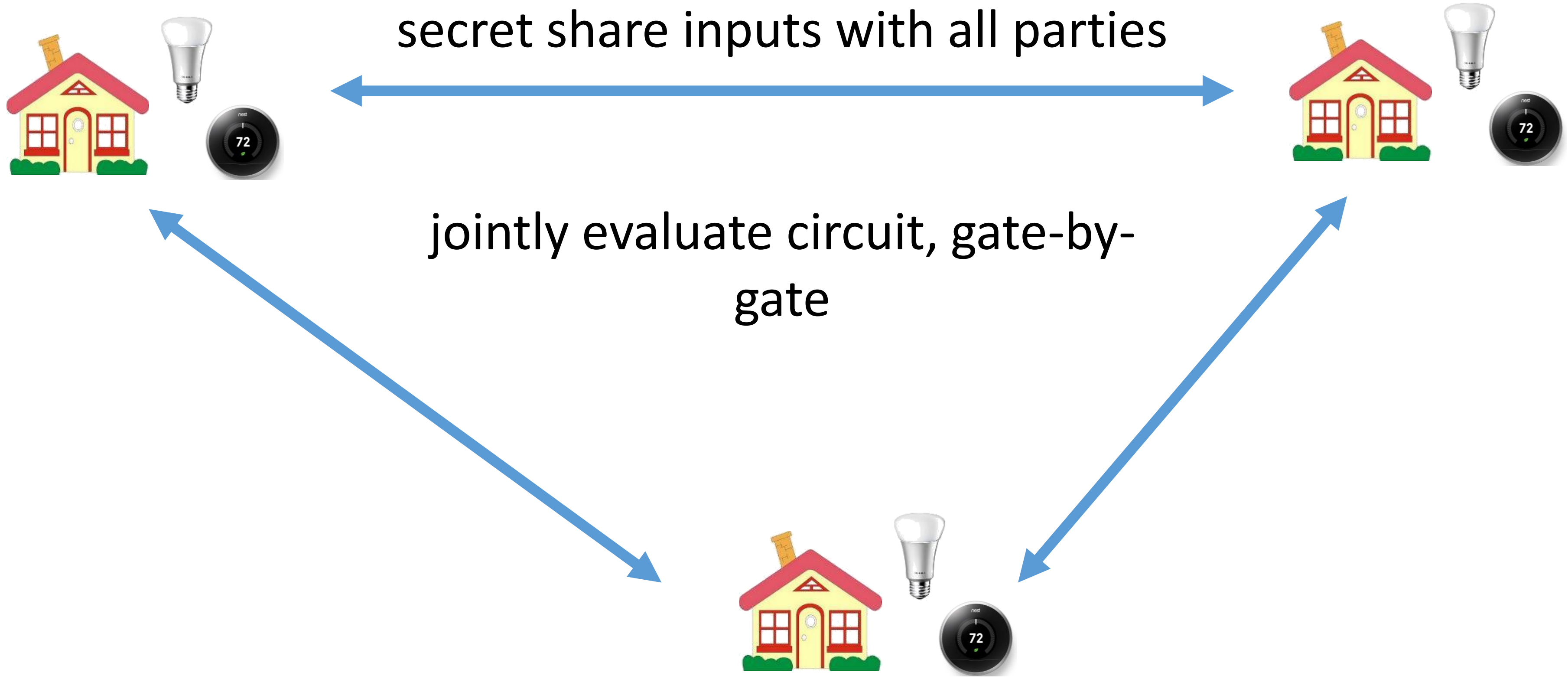
Two-Party Computation (2PC)

- Yao's circuits very efficient and heavily optimized [KSS09]
  - Evaluating circuits with 1.29 ***billion*** gates in 18 minutes (1.2 gates / μs) [ALSZ13]

- Yao's circuit provides semi-honest security: malicious security via cut-and-choose, but not as efficient

Going Beyond 2PC

- General MPC also “solved” [GMW87]



Secure Multiparty Computation

- General MPC suffices to evaluate arbitrary functions amongst many parties: should be viewed as a <u>feasibility</u> result
- Limitations of general MPC
  - many rounds of communication / interaction
  - possibly large bandwidth
  - hard to coordinate interactions with large number of parties
- Other considerations (not discussed): fairness, guaranteeing output delivery

This Talk: Homomorphic Encryption

Many rounds of interaction
Boolean circuits (typically)

Few rounds of interaction
Arithmetic circuits

GMW Protocol and
General MPC

Custom Protocols

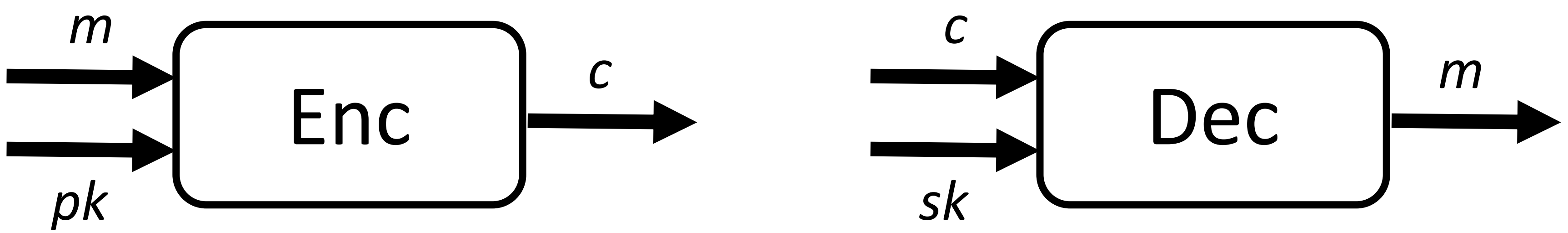
Homomorphic
Encryption

General methods for secure computation

Homomorphic Encryption

Homomorphic encryption scheme: encryption scheme that allows computation on ciphertexts

Comprises of three functions:

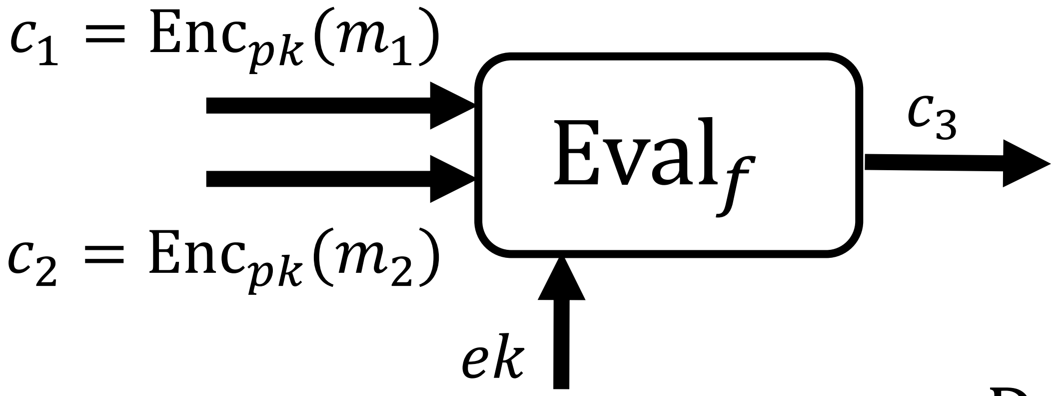


Must satisfy usual notion of semantic security

Homomorphic Encryption

Homomorphic encryption scheme: encryption scheme that allows computation on ciphertexts

Comprises of three functions:



$$\mathrm{Dec}_{sk}\left(\mathrm{Eval}_f(ek, c_1, c_2)\right) = f(m_1, m_2)$$

Fully Homomorphic Encryption (FHE)

Many homomorphic encryption schemes:

- ElGamal: $f(m_0, m_1) = m_0 m_1$
- Paillier: $f(m_0, m_1) = m_0 + m_1$

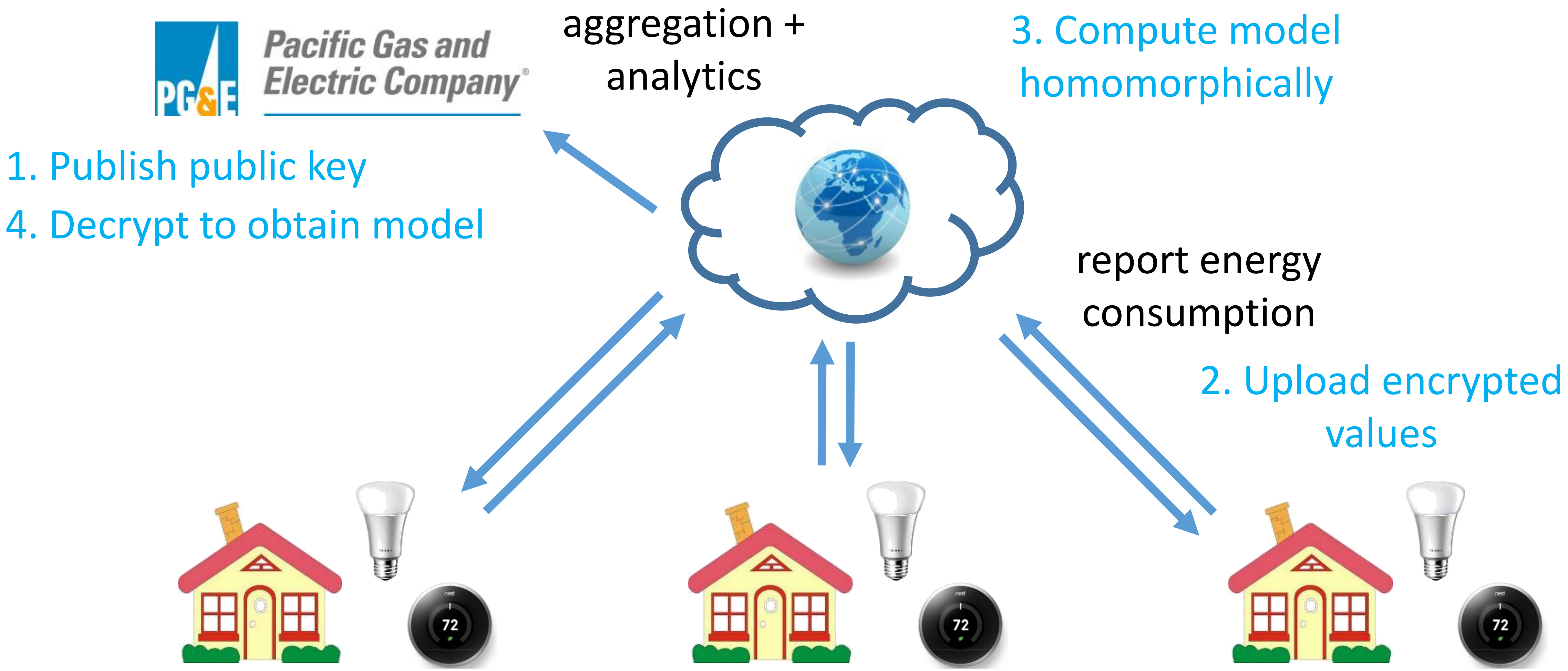
Fully homomorphic encryption: homomorphic with respect to **two** operations: addition and multiplication

- [BGN05]: one multiplication, many additions
- [Gen09]: first FHE construction from lattices

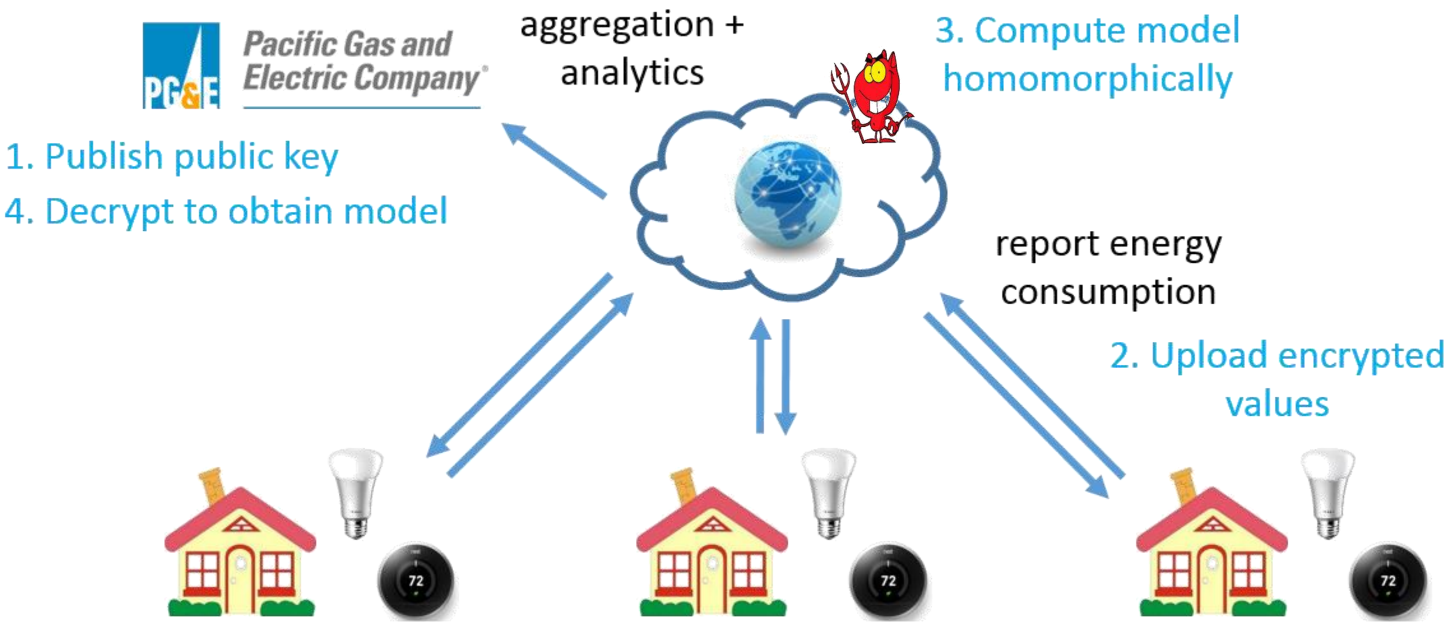
Privately Outsourcing Computation

Leveraging computational power of the cloud

Machine Learning in the Cloud



Machine Learning in the Cloud



- Passive adversary sitting in the cloud does *not* see client data
- Power company only obtains resulting model, not individual data points (assuming no collusion)
- Parties only need to communicate with cloud (the power of public-key encryption)

Big Data, Limited Computation

- Homomorphic encryption is expensive, especially compared to symmetric primitives such as AES
- Can be unsuitable for encrypting large volumes of data

"Hybrid" Homomorphic Encryption

$\text{Enc}_{pk}(k), \text{AES}_k(m)$

Homomorphically evaluate the AES decryption circuit

Encrypt AES key using homomorphic encryption (expensive), encrypt data using AES (cheap)

$\text{AES}_k(m)$ → encrypt → $\text{Enc}_{pk}(\text{AES}_k(m))$

evaluate AES decryption

$\text{Enc}_{pk}(k)$ → $\text{Enc}_{pk}(m)$

homomorphic evaluation

$\text{Enc}_{pk}(f(m))$

Current performance: ≈ 400 seconds to decrypt 120 AES-128 blocks (4 s/block) [GHS15]

Constructing FHE

- FHE: can homomorphically compute arbitrary number of operations

- Difficult to construct – start with something simpler: *somewhat homomorphic encryption scheme (SWHE)*

- SWHE: can homomorphically evaluate a few operations (circuits of low depth)

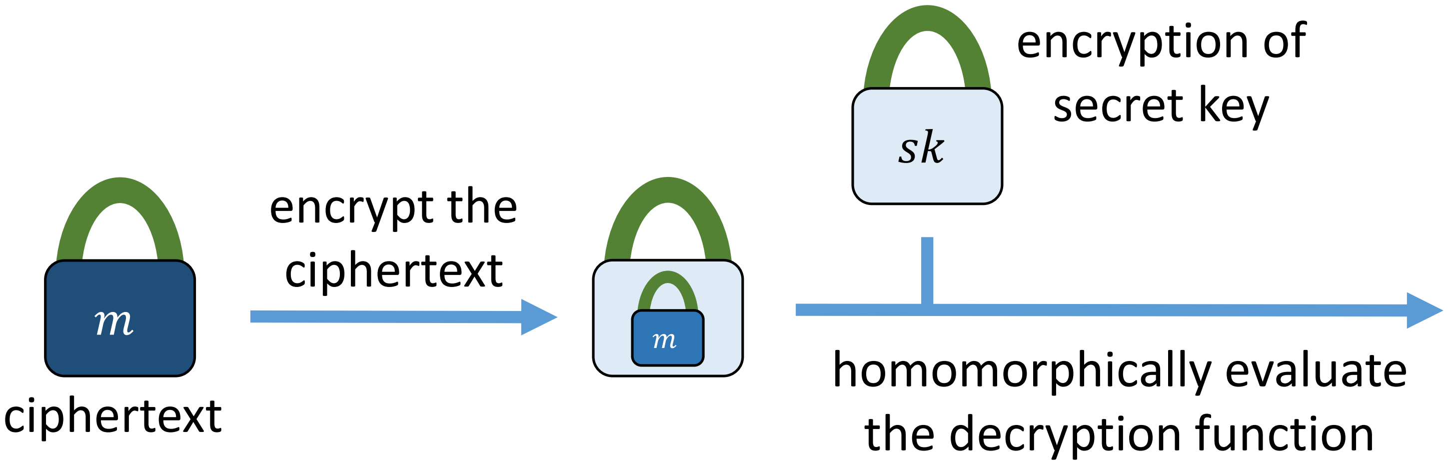
Gentry's Blueprint: SWHE to FHE

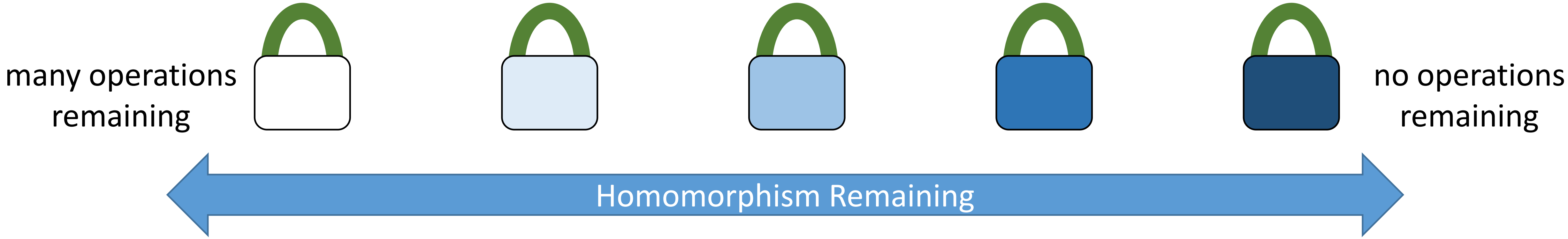
- Gentry described general *bootstrapping* method of achieving FHE from SWHE [Gen'09]

- Starting point: SWHE scheme that can evaluate its own decryption circuit

Gentry’s Blueprint: From SWHE to FHE

recrypt functionality





Bootstrappable SWHE

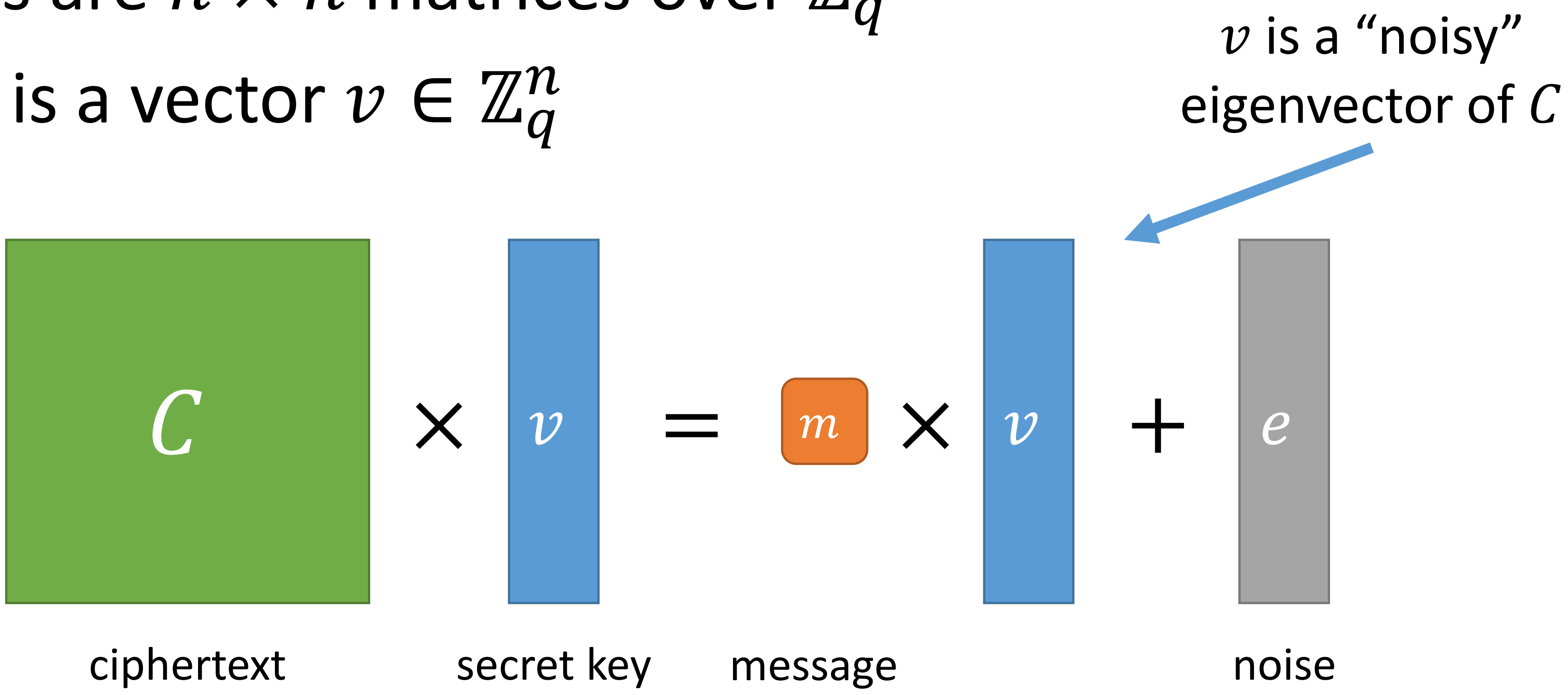
- First bootstrappable construction by Gentry based on ideal lattices [Gen09]

- Tons of progress in constructions of FHE in the ensuing years [vDGHV10, SV10, BV11a, BV11b, Bra12, BGV12, GHS12, GSW13], and more!

- Have been simplified enough that the description can fit in a blog post [BB12]

Conceptually Simple FHE [GSW13]

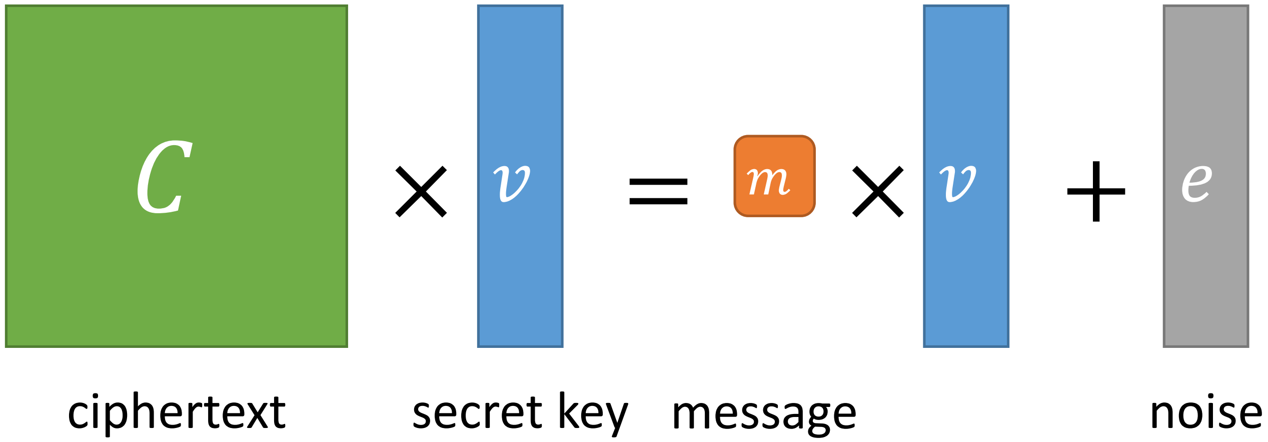
- Ciphertexts are $n \times n$ matrices over $\mathbb{Z}_q$
- Secret key is a vector $v \in \mathbb{Z}_q^n$



Encryption of m satisfies above relation

Conceptually Simple FHE [GSW13]

- Suppose that v has a "large" component v_i



- Can decrypt as follows:

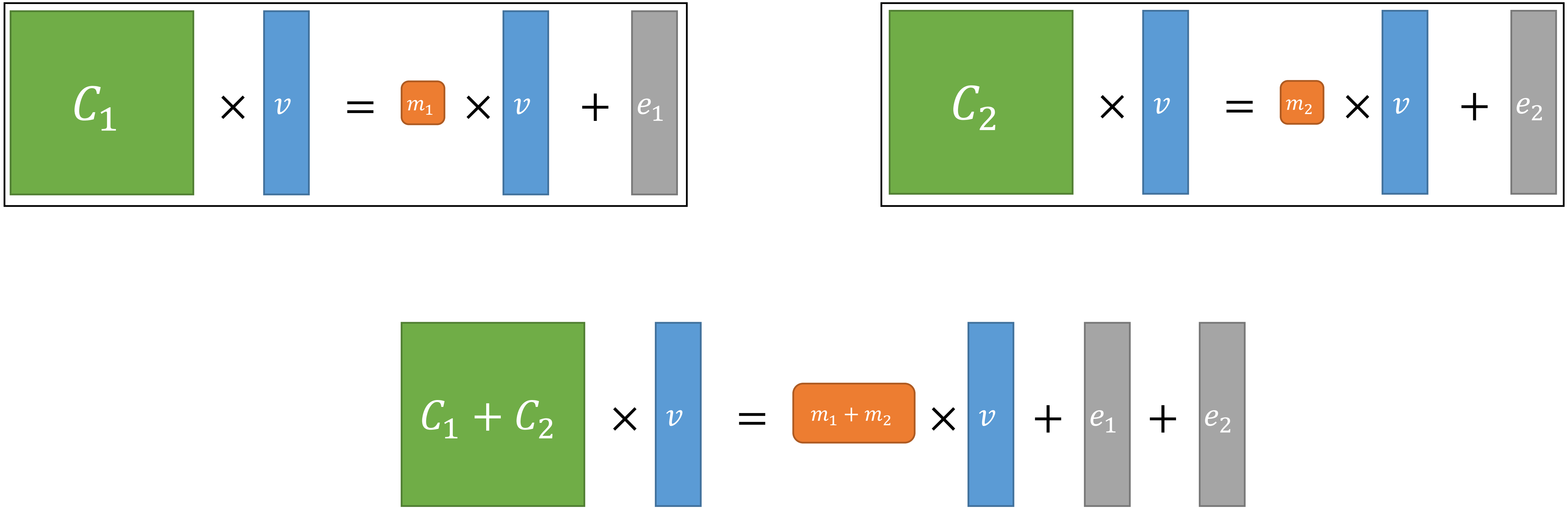
$$\left\lfloor \frac{\langle C_i, v \rangle}{v_i} \right\rceil = \left\lfloor \frac{m v_i + e_i}{v_i} \right\rceil = m$$

C_i is i^{th} row of C

Relation holds if $\left|\frac{e_i}{v_i}\right| < \frac{1}{2}$

Conceptually Simple FHE [GSW13]

Homomorphic addition

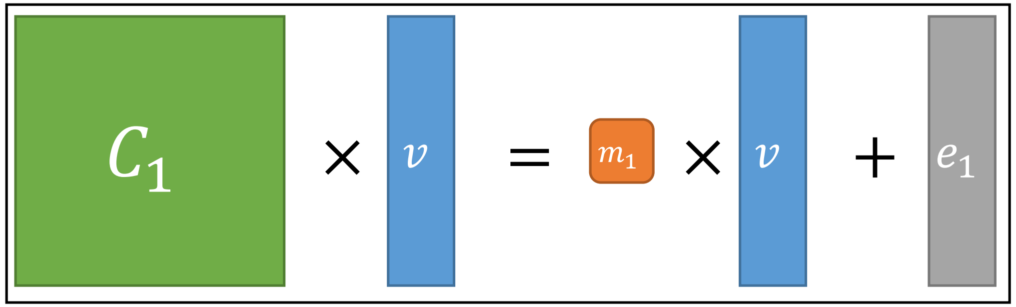


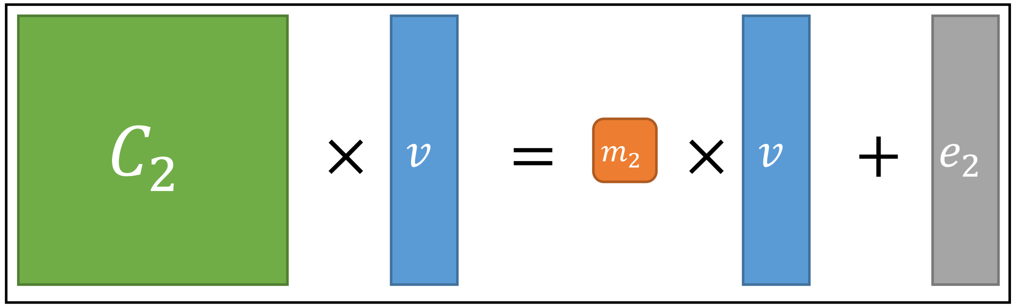
homomorphic addition is
matrix addition

noise terms also add

Conceptually Simple FHE [GSW13]

Homomorphic multiplication





$$(C_1 C_2)v = (m_1 m_2)v + C_1 e_2 + m_2 e_1$$

homomorphic multiplication
is matrix multiplication

noise could blow up if
C_1 or m_2 are not small

Conceptually Simple FHE [GSW13]

- Basic principles: ciphertexts are matrices, messages are approximate eigenvalues

- Homomorphic operations correspond to matrix addition and multiplication (and some tricks to constrain noise)

- Hardness based on learning with errors (LWE) [Reg05]

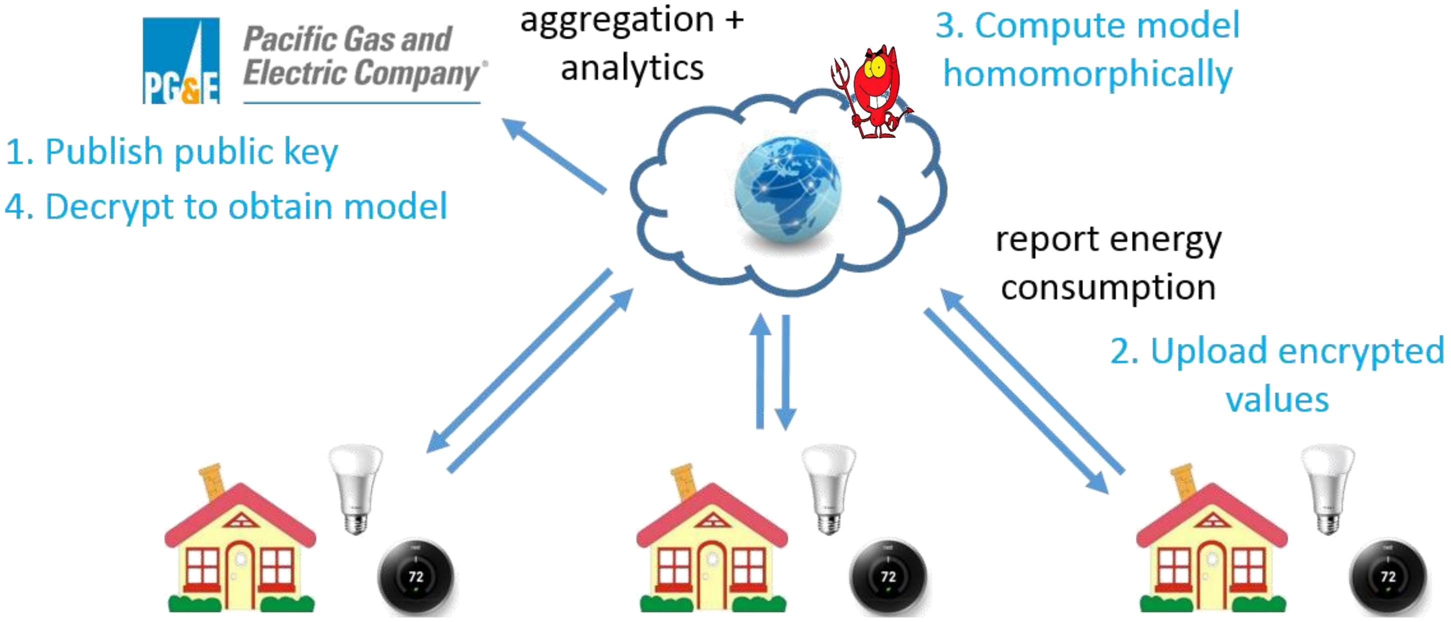
The Story so Far…

- Simple FHE schemes exist

- But… bootstrapping is expensive!
  - At 76 bits of security: each bootstrapping operation requires 320 seconds and 3.4 GB of memory [HS14]
  - Other implementations exist, but generally less flexible / efficient

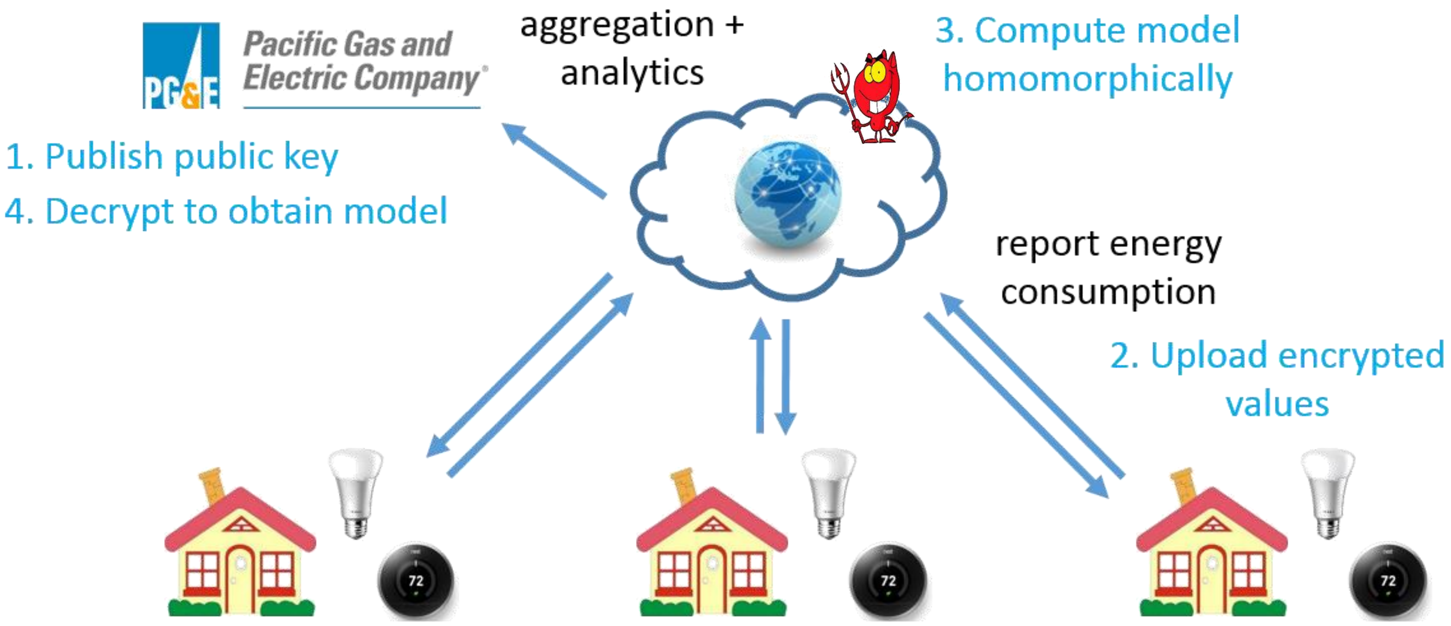
- SWHE (without bootstrapping) closer to practical: can evaluate shallow circuits

Application: Statistical Analysis



- Consider simple statistical models: computing the mean or covariance (for example, average power consumption)
- Problem: given n vectors $x_1, \dots, x_n$, compute
  - Mean: $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$
  - Covariance: $\Sigma_X = \frac{1}{n^2} (nX^T X -$

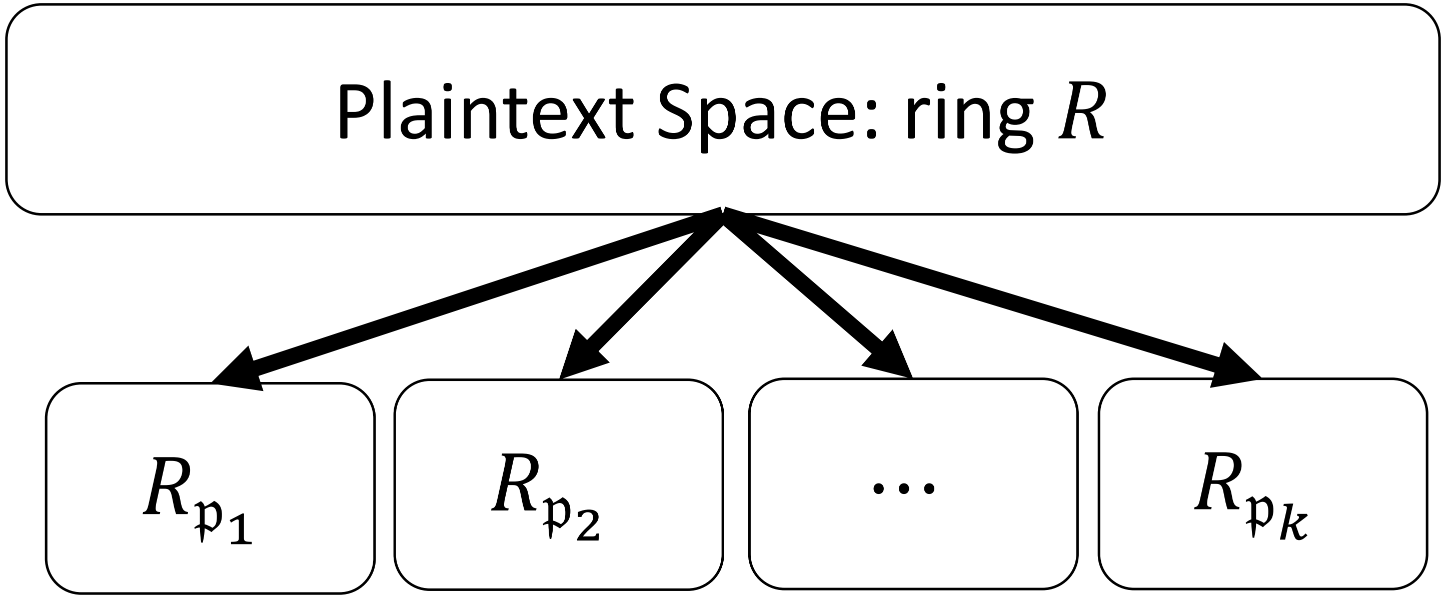
Application: Statistical Analysis



- Can also perform linear regression: given design matrix X and response vector y, evaluate normal equations
$$\theta = (X^T X)^{-1} X^T y$$
- Matrix inversion (over $\mathbb{Q}$) using Cramer's rule
- Depth n for n-dimensional data

Batch Computation [SV11]

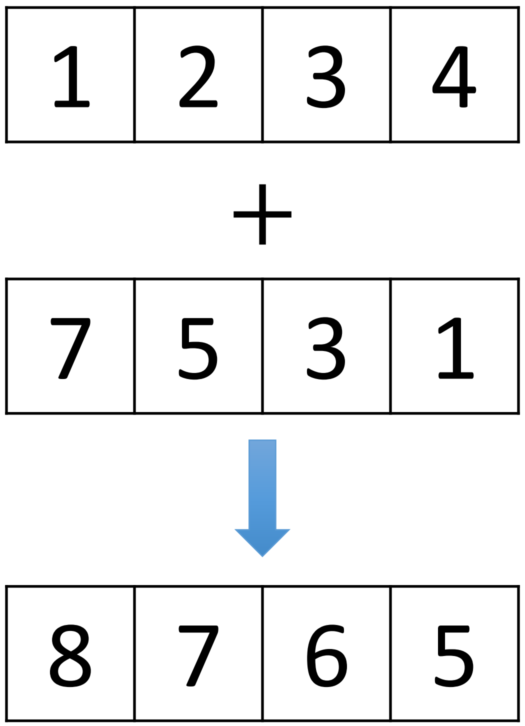
Algebraic structure of some schemes enable encryption + operations on vectors at no extra cost



Chinese Remainder Theorem: $R \cong \bigotimes_{i=1}^{k} R_{\mathfrak{p}_i}$

Batch Computation [SV11]

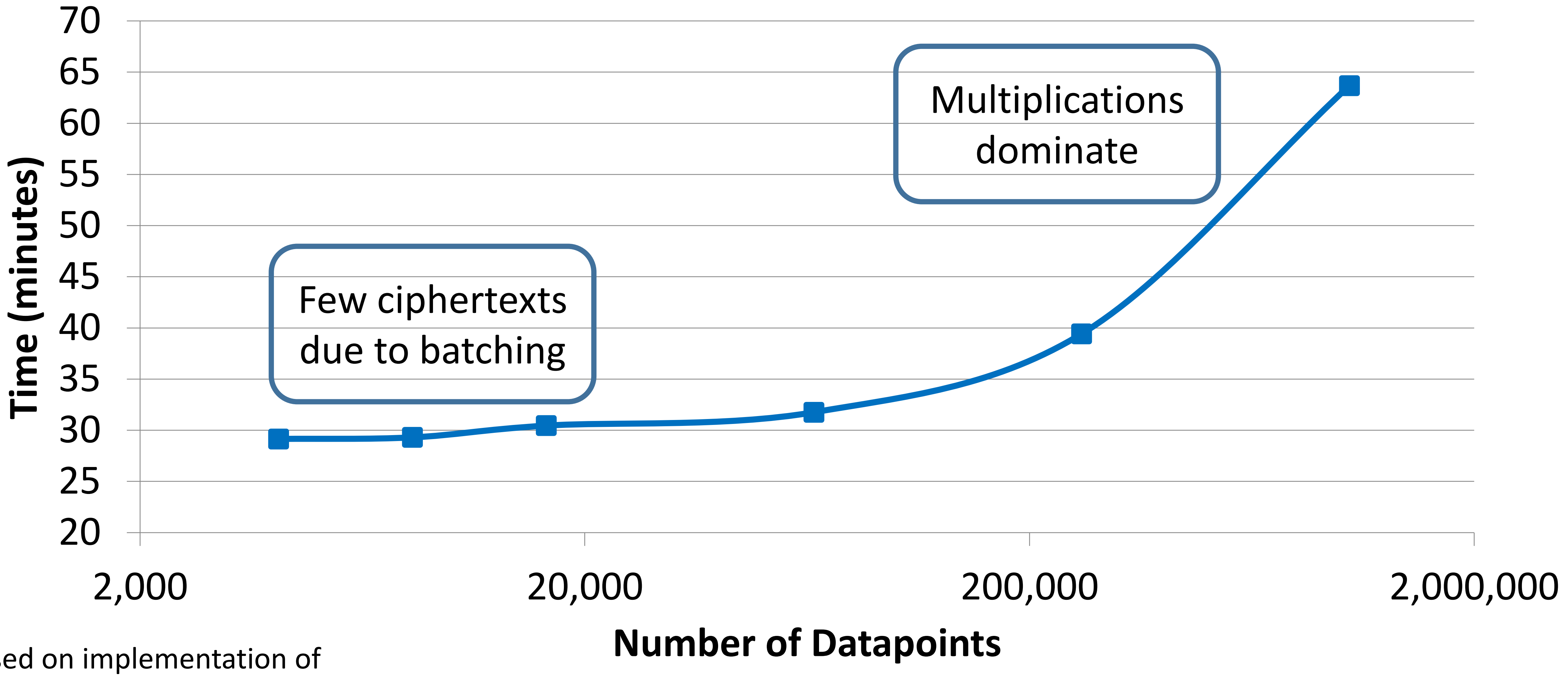
Encrypt + process array of values at no extra cost:



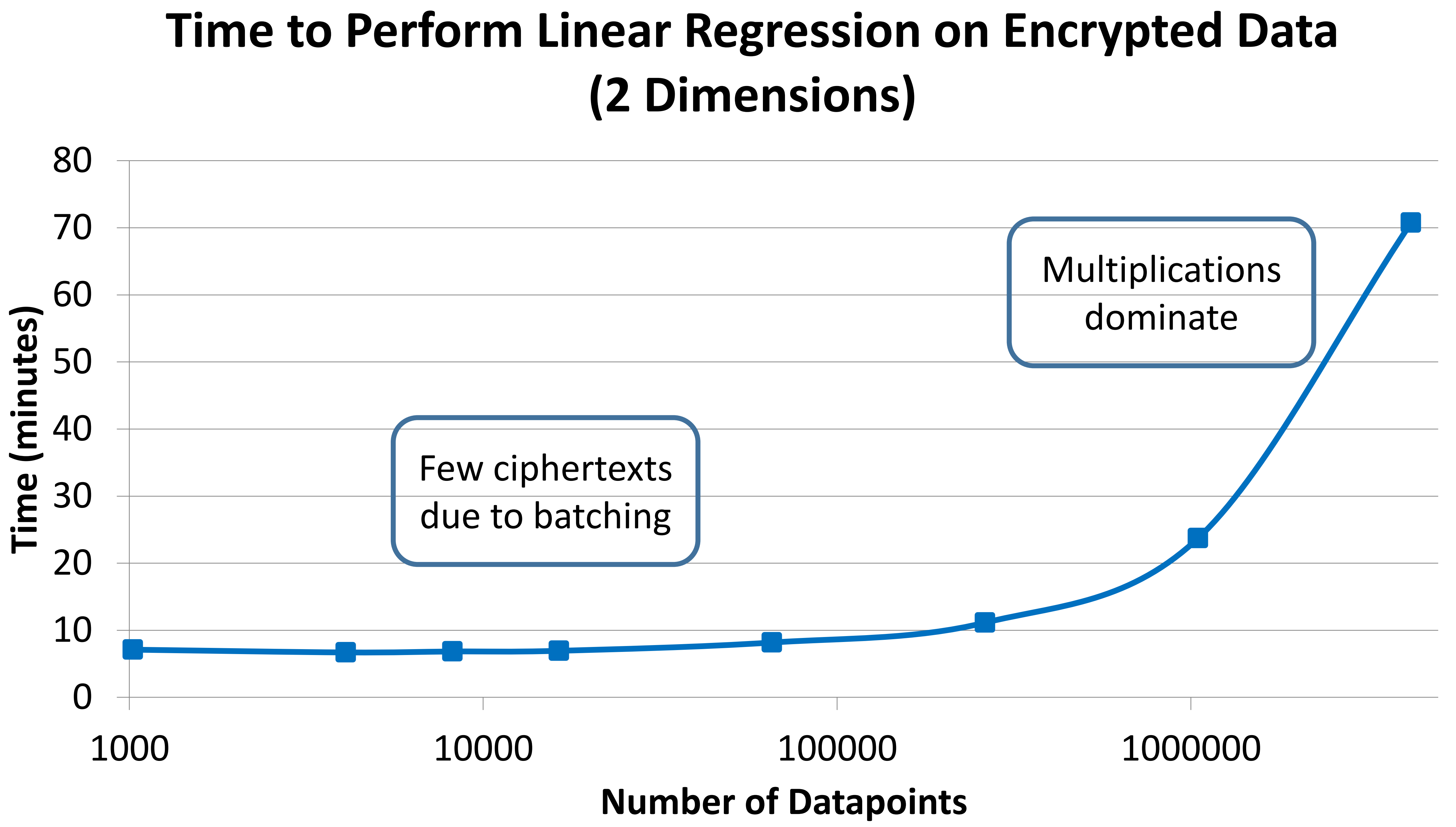
One homomorphic operation

In practice: ≥ 5000 slots

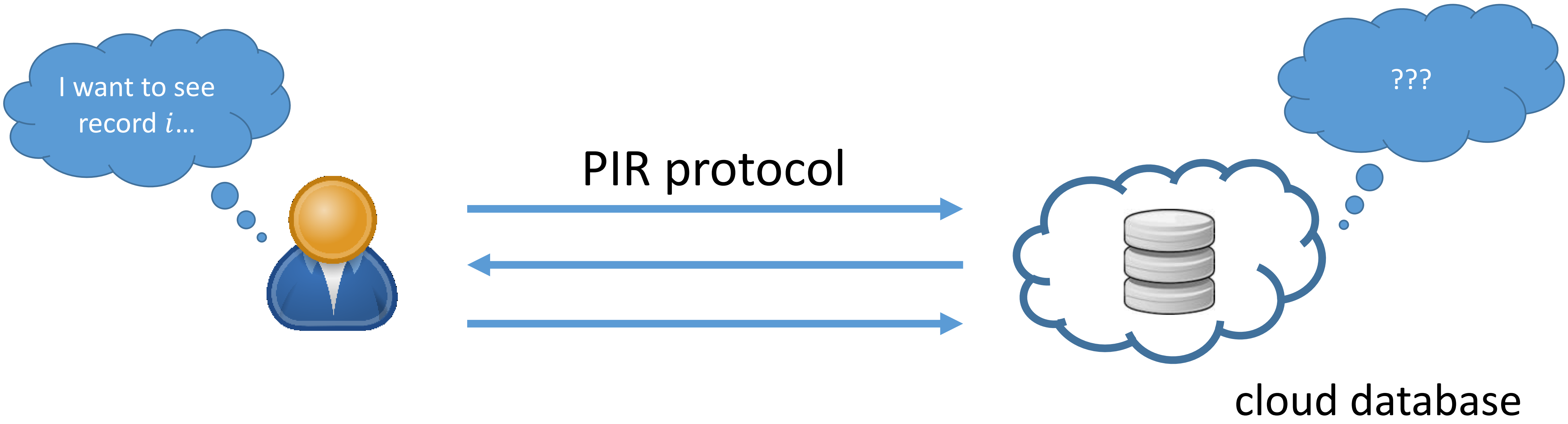
Time to Compute Mean and Covariance over Encrypted Data (Dimension 4)



Based on implementation of Brakerski's scheme [Bra12]


Time to Perform Linear Regression on Encrypted Data
(2 Dimensions)
Time (minutes)
80
70
60
50
40
30
20
10
0
1000
10000
100000
1000000
Number of Datapoints
Few ciphertexts due to batching
Multiplications dominate

Application: Private Information Retrieval



client learns record i, server learns nothing

PIR from Homomorphic Encryption [KO97]

$$\begin{bmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} v_{11} \\ v_{21} \\ v_{31} \end{bmatrix}$$

represent database as matrix

query is an encrypted basis vector

response

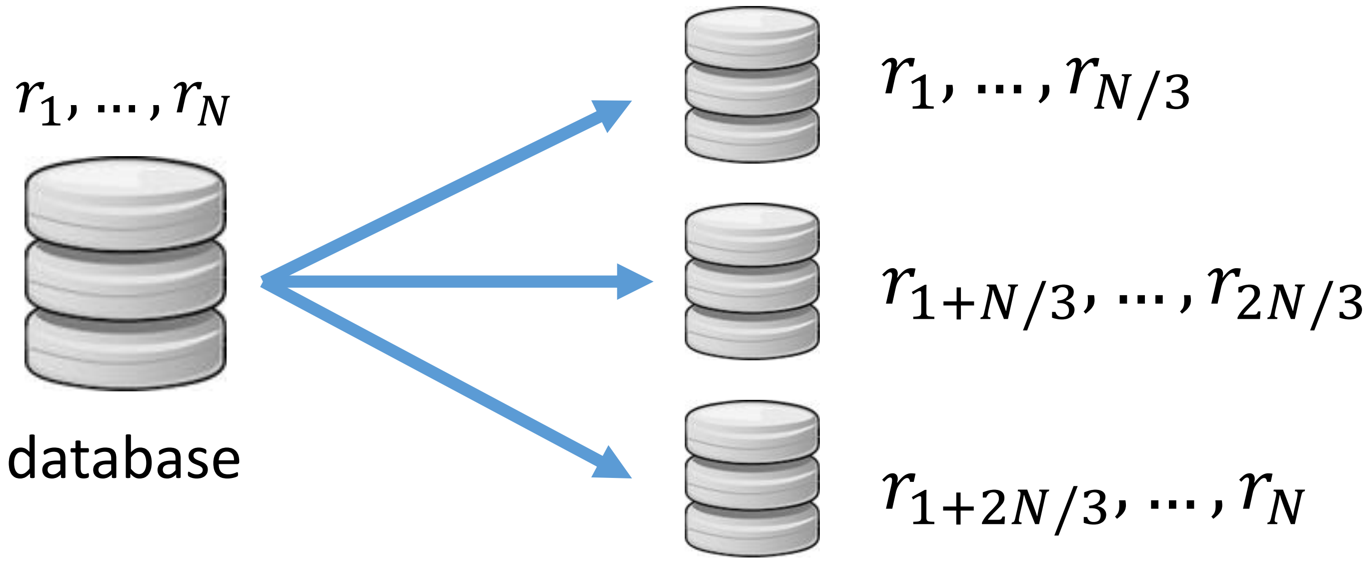
$O(\sqrt{n})$ communication

server evaluates inner product

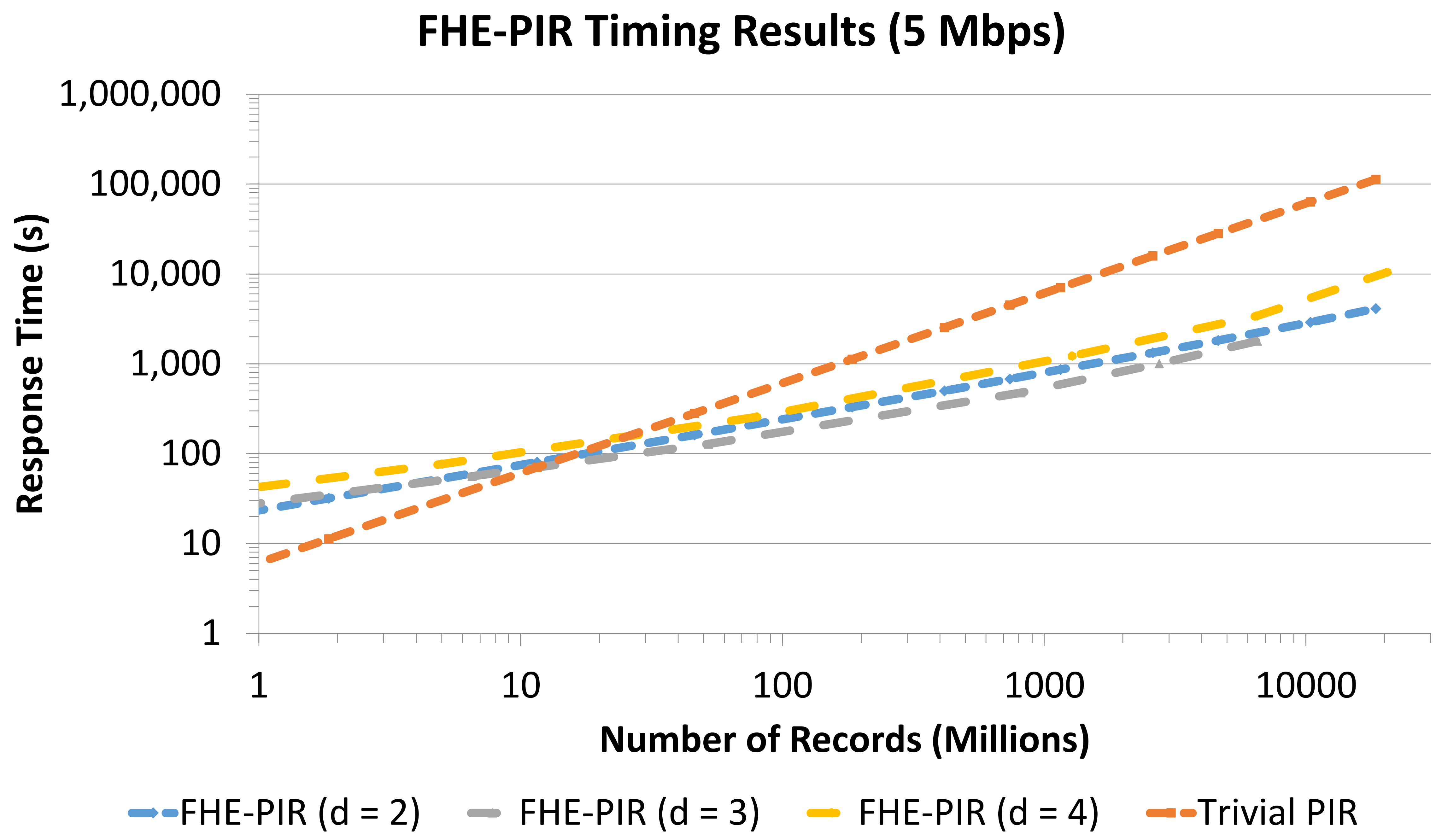
database components in the clear: additive homomorphism suffices

PIR from Homomorphic Encryption

- $O(\sqrt{n})$ communication with additive homomorphism alone
- Naturally generalizes:
  - $O(\sqrt[3]{n})$ with one multiplication
  - $O\left(\sqrt[k]{n}\right)$ with degree $(k-1)$-homomorphism
- Benefits tremendously from batching



split database into many small databases, query in parallel


FHE-PIR Timing Results (5 Mbps)
Response Time (s)
1,000,000
100,000
10,000
1,000
100
10
1
1
10
100
1000
10000
Number of Records (Millions)
FHE-PIR (d = 2)
FHE-PIR (d = 3)
FHE-PIR (d = 4)
Trivial PIR

PIR from Homomorphic Encryption

- Outperforms trivial PIR for very large databases

- However, recursive KO-PIR with additive homomorphism is still state-of-the-art

Concluding Remarks

- Internet of Things brings many security challenges
- Many generic cryptographic tools: 2PC, MPC, FHE
  - 2PC/MPC work well for small number of parties
  - SWHE/FHE preferable with many parties (IoT scale)
- FHE still nascent technology – should be viewed as a “proof-of-concept” rather than practical solution
- SWHE closer to practical, suitable for evaluating simple (low-depth) functionalities
- Big open problem to develop more practical constructions!

Questions?