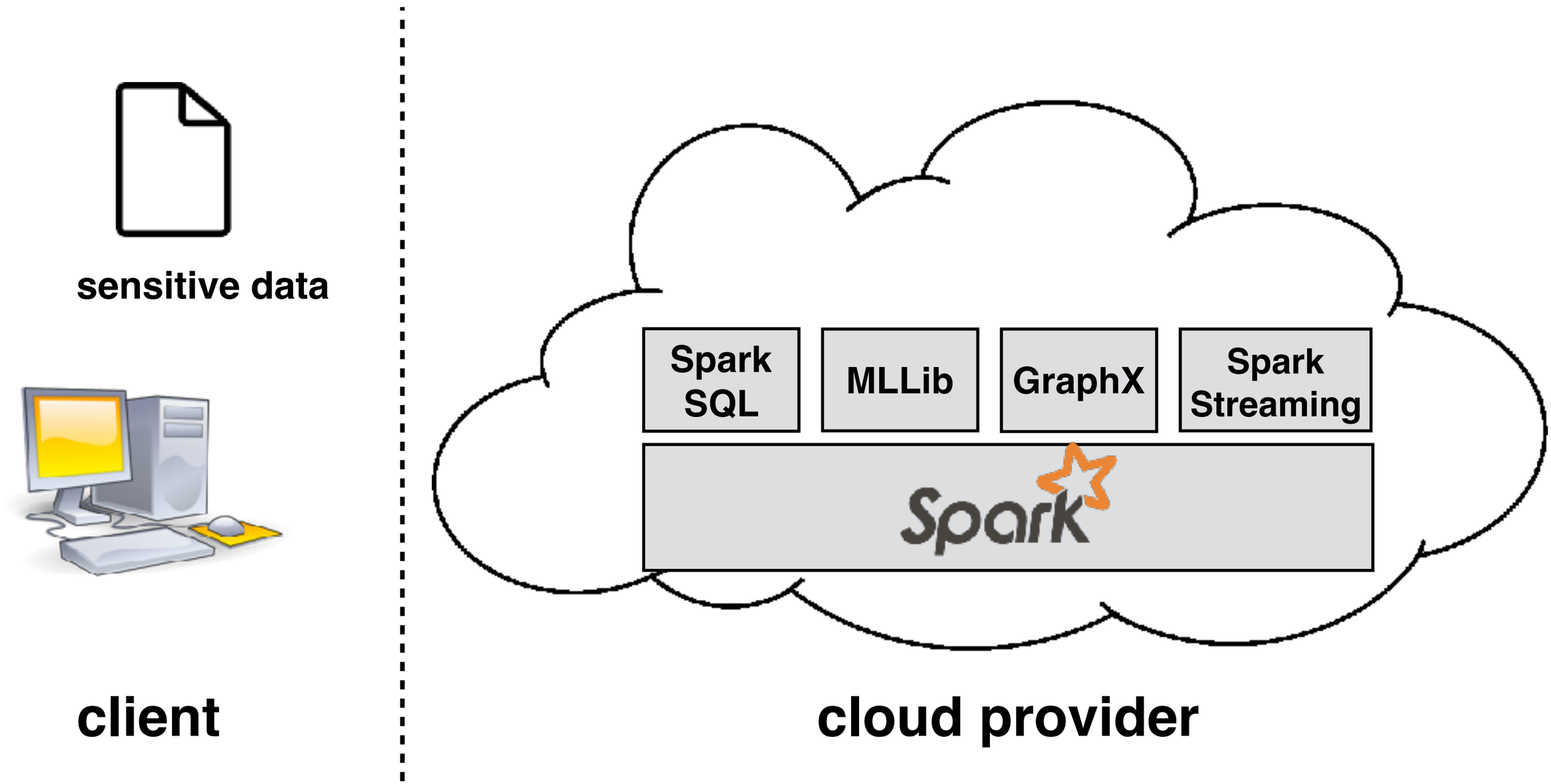
Opaque: An Oblivious and Encrypted Distributed Analytics Platform

[NSDI'17]

Joint work with: Wenting Zheng, Ankur Dave, Jethro Beekman, Raluca Ada Popa, Joseph Gonzalez, and Ion Stoica

UC Berkeley

Complex analytics run on sensitive data



NEWS ANALYSIS

No, your data isn't secure in the cloud

In 2012, Google alone received 21,389 government requests for information affecting 33,634 user accounts

By Lucas Mearian

Senior Reporter, Computerworld | AUG 13, 2013 7:00 AM PT

Safe Computing | 09 Apr 2012 | Print | Comment (7)

How to Tell If Your Cloud Provider Can Read Your Data

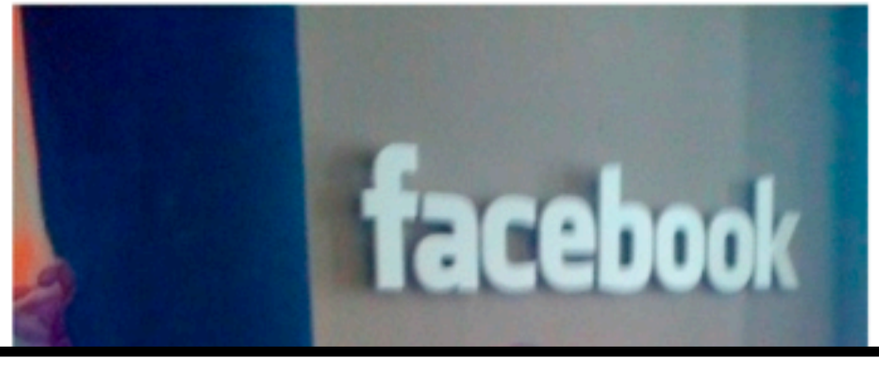
by Rich Mogull

With the tremendous popularity of services like Dropbox and iCloud there is, rightfully, an incredible amount of interest in cloud data security. Once we start hosting our most sensitive data with cloud services (or any third-party provider) it's only natural to wonder how secure our data is when it's in the hands of others. But sometimes it's hard to figure out exactly *who* can look at our information, especially since buzzwords like "secure" and "encrypted" don't necessarily mean *you*

Why You Shouldn't Trus
Your Data: An Employee

Ryan Tate

01/11/10 02:34PM Filed to: YOUR PRIVACY IS AN ILLUSION



The abuse o
employees
simple act o
corruption.
social netw

data i
t the
ch er
priva

huge

age

Sony pays high price for ignoring its past

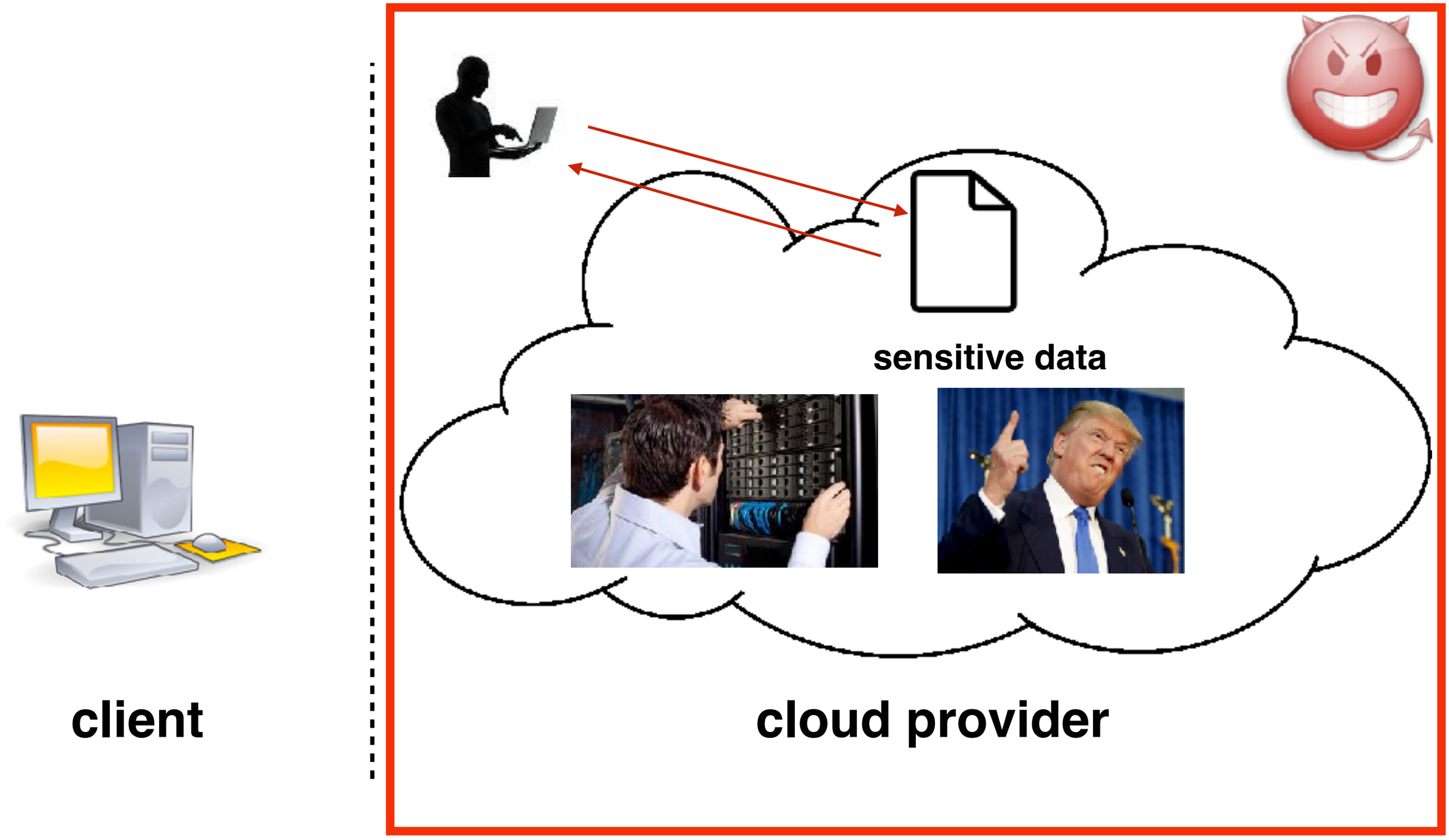
Dec. 19, 2014

The world's biggest data breaches 2015 - 888 incidents, 246 million records, uncounted misery

By John E Dunn | Sep 09, 2015

Share

Cloud attackers



Threat model

Attacker has full access to all cloud software

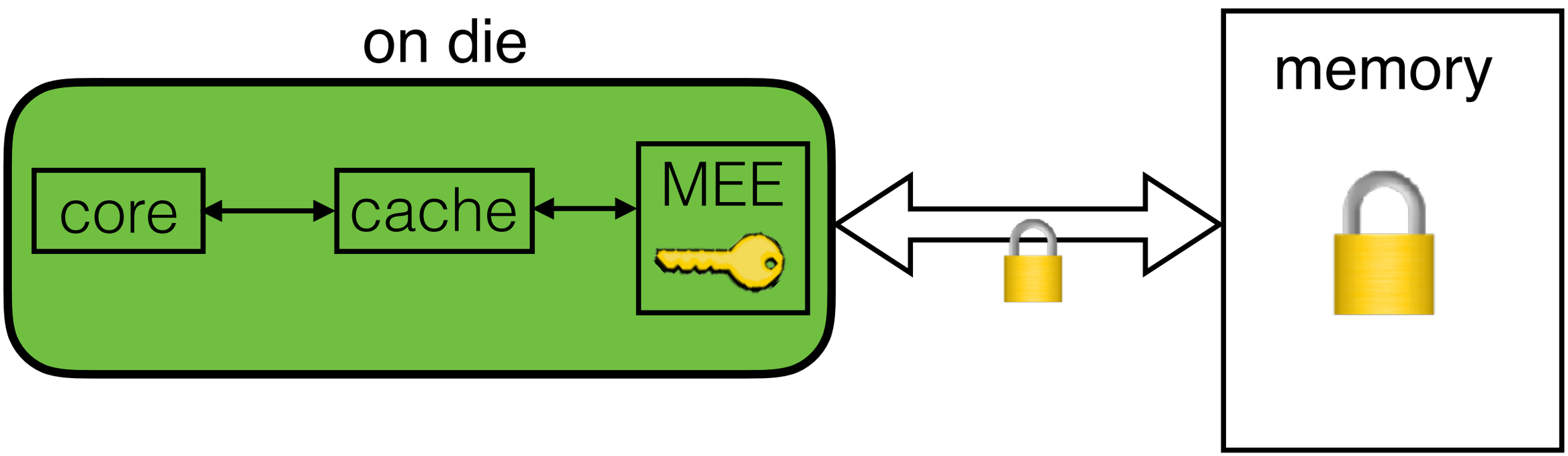
How to protect
data and computation
while preserving functionality?

relational algebra

Hardware enclaves 101

Hardware enclaves (Intel SGX)

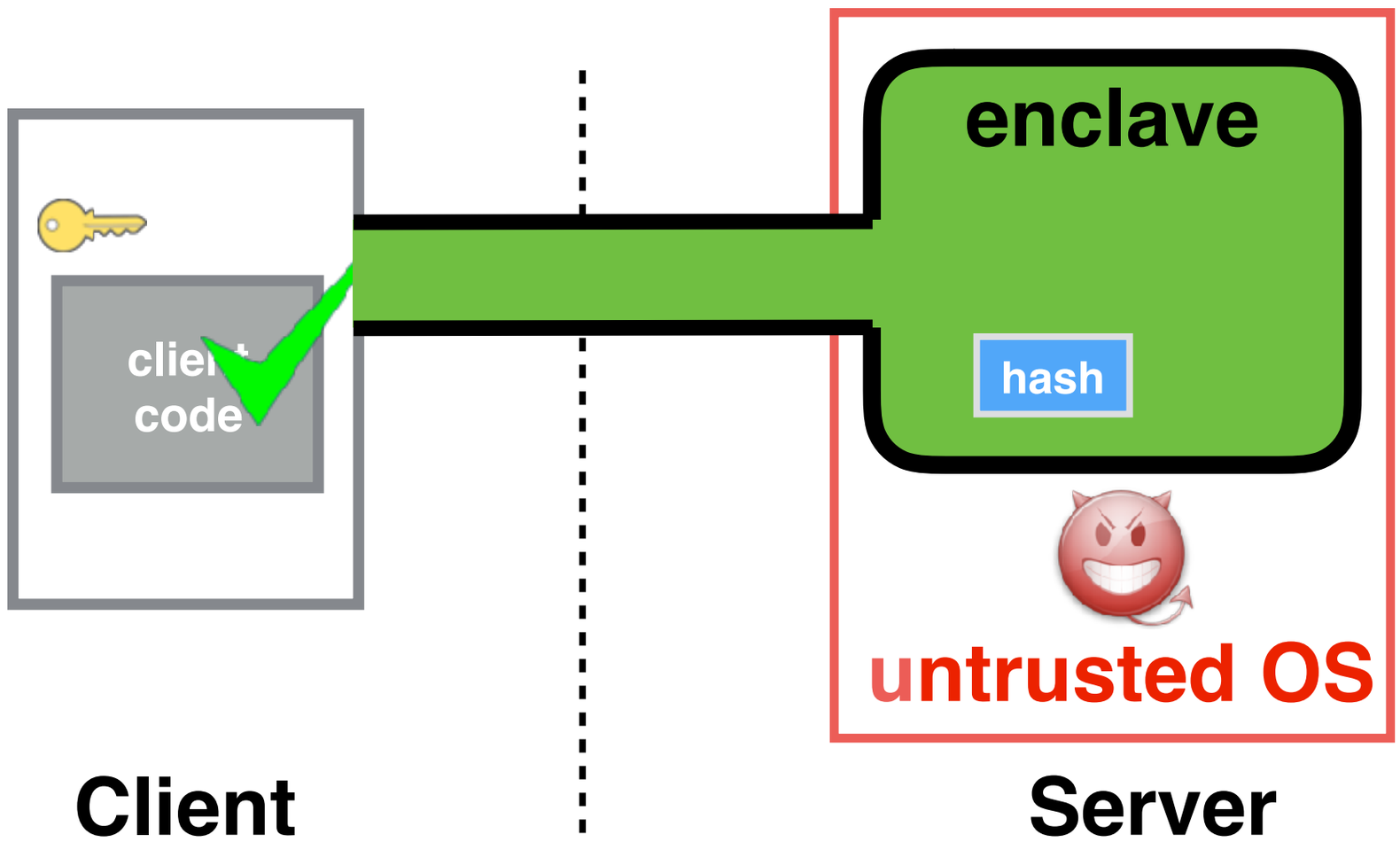
- Hardware-enforced isolated execution environment
- Data decrypted only on the processor



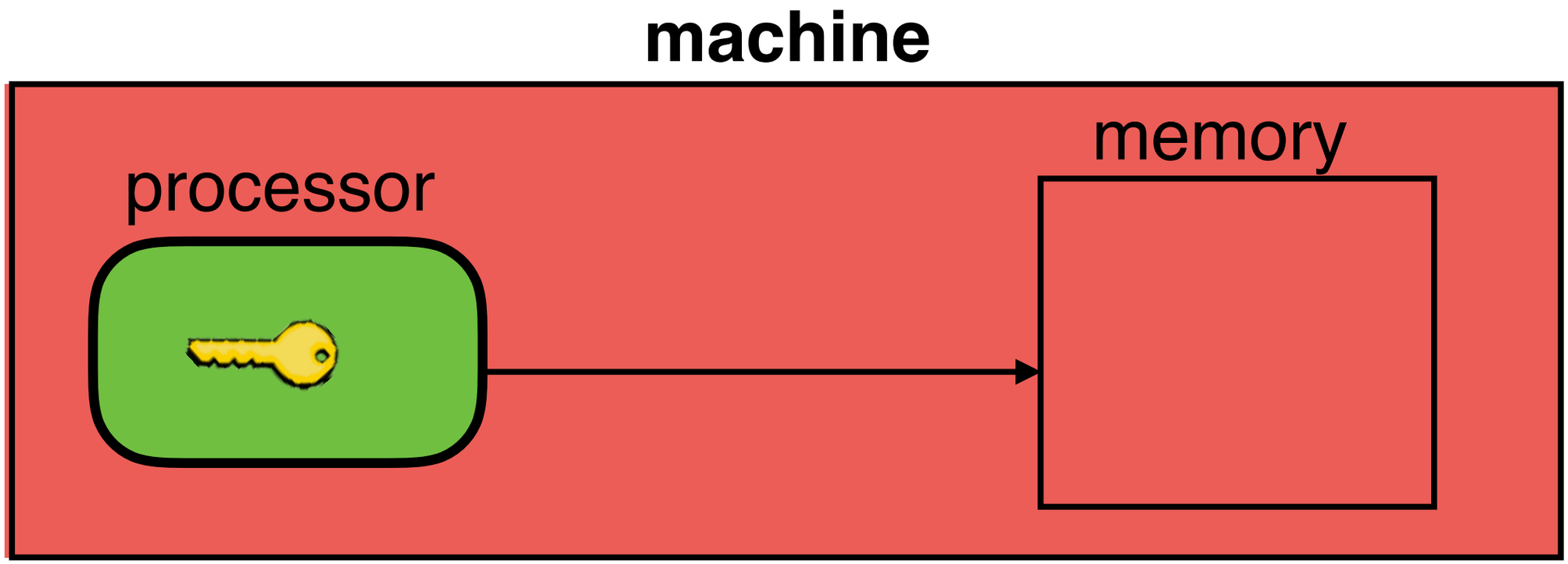
- Protect against an attacker who has root access or compromised OS

Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



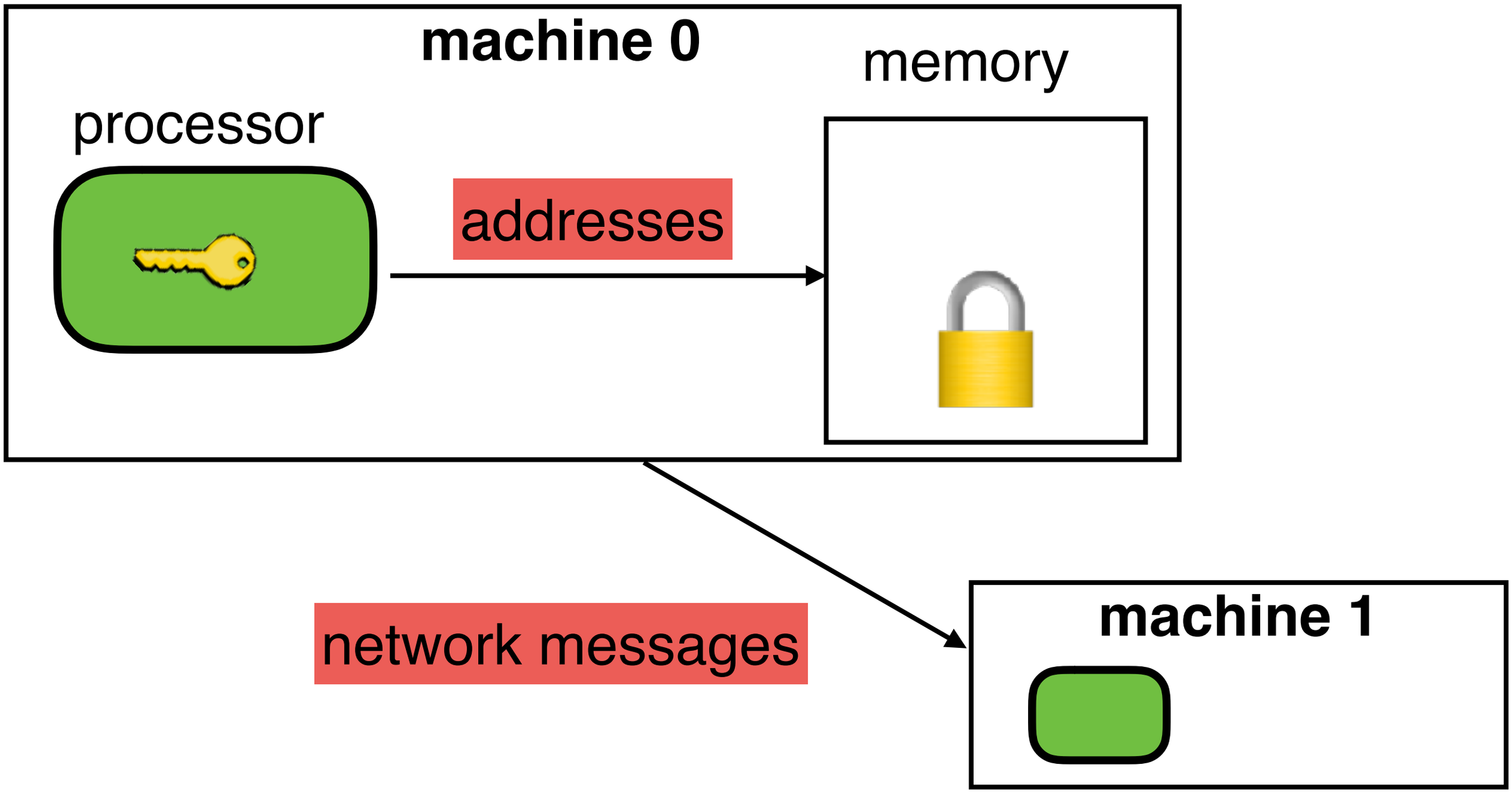
Attacker model



Attacker is restricted to software attacks only, and does not exploit timing. Attacker controls the software stack.

Access patterns leakage

Access patterns

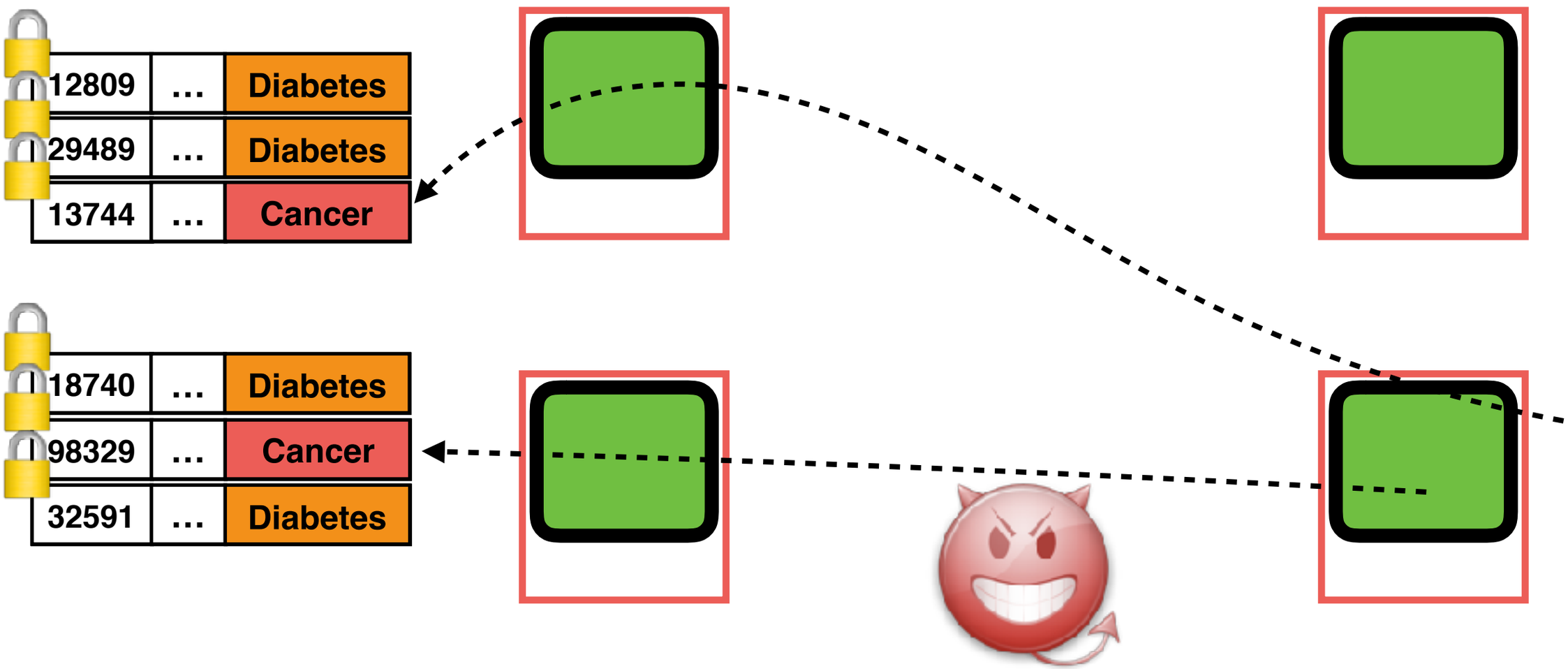


Example: network access pattern leakage

ID	Name	Age	Disease
12809	Amanda D. Edwards	40	Diabetes
29489	Robert R. McGowan	56	Diabetes
13744	Kimberly R. Seay	51	Cancer
18740	Dennis G. Bates	32	Diabetes
98329	Ronald S. Ogden	53	Cancer
32591	Donna R. Bridges	26	Diabetes

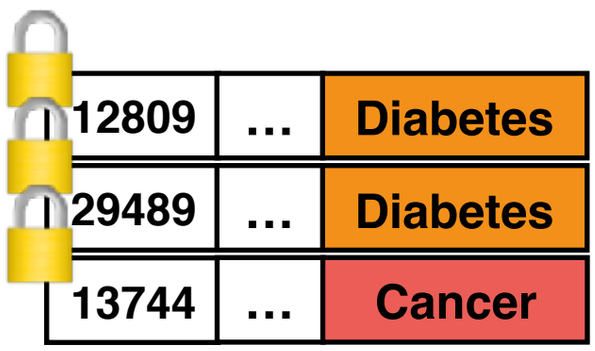
SELECT count(*) FROM medical
GROUP BY disease

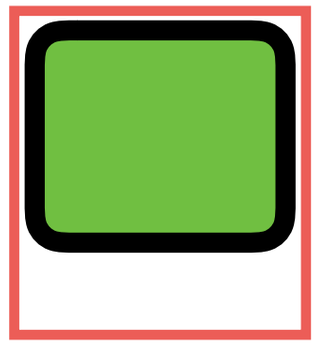
Example: network access pattern leakage

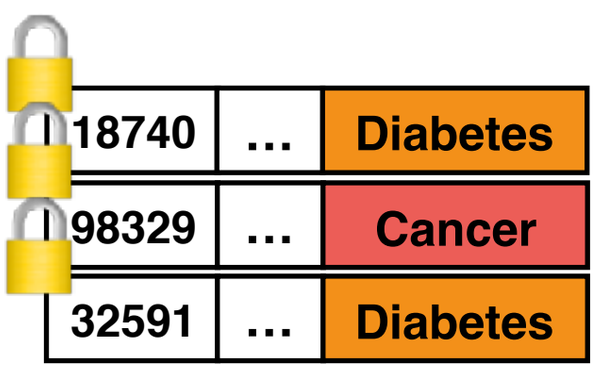


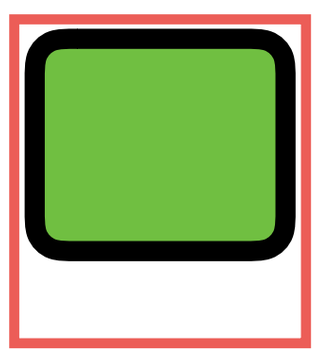
Public information:
Diabetes twice as common as cancer

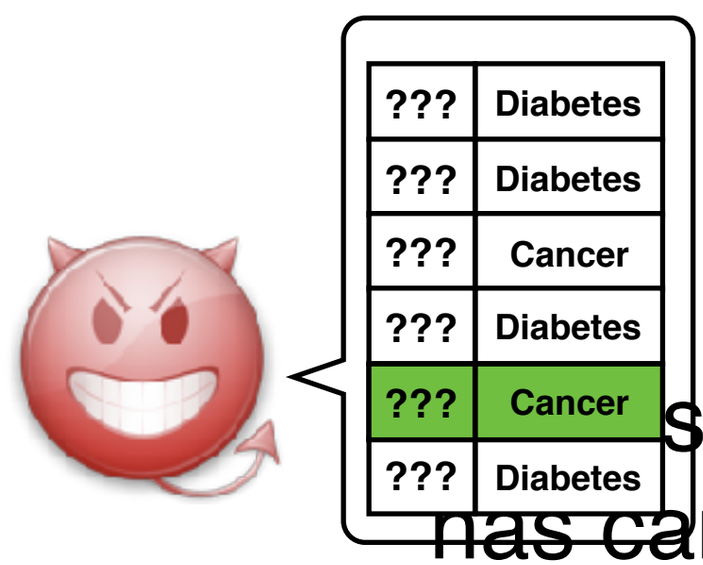
Example: network access pattern leakage











s that Alice
has cancer

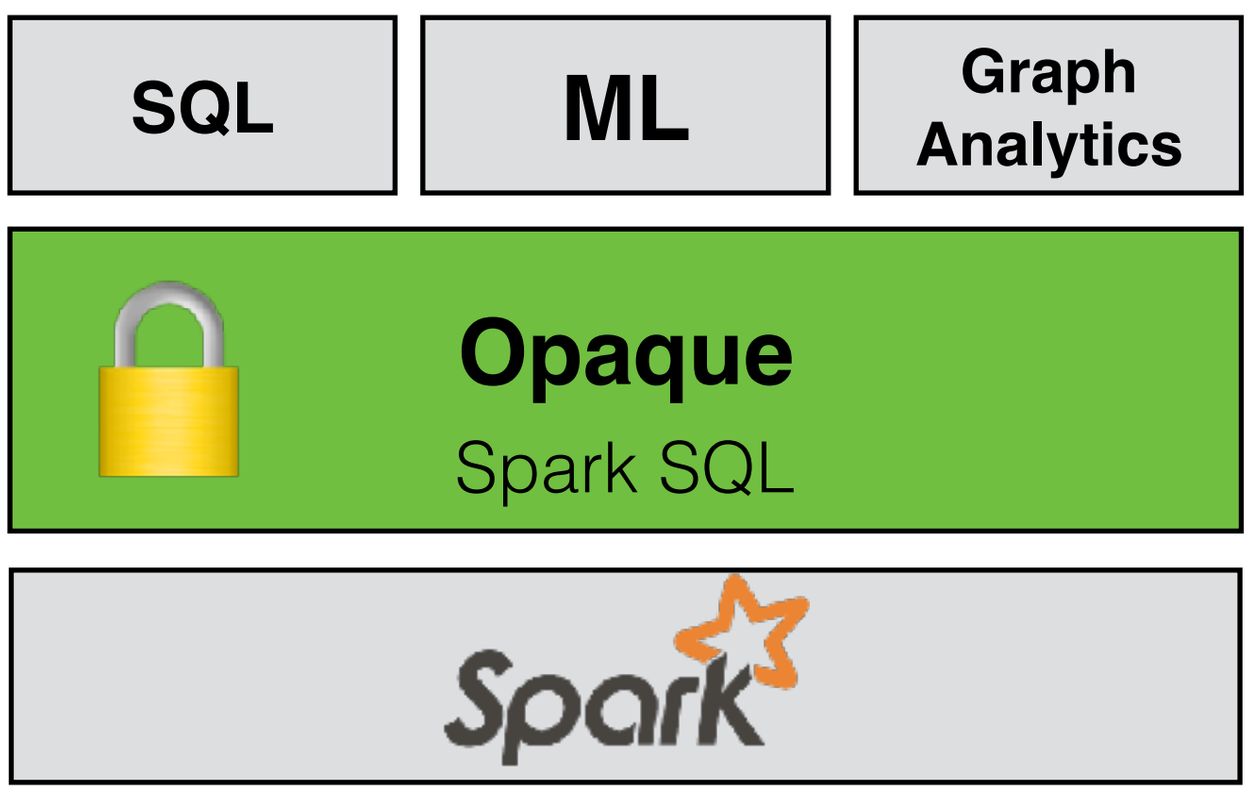
Leakage from prior work

- Memory access patterns attacks [XCP15] extracted complete text documents and photo outlines
- Network access patterns [OCF+15] extracted age, gender, address of individuals

access patterns are independent of data content

Goal: oblivious distributed analytics

Opaque*: oblivious and encrypted distributed analytics platform



* Oblivious Platform for Analytic QUEries

Security guarantees (informal)

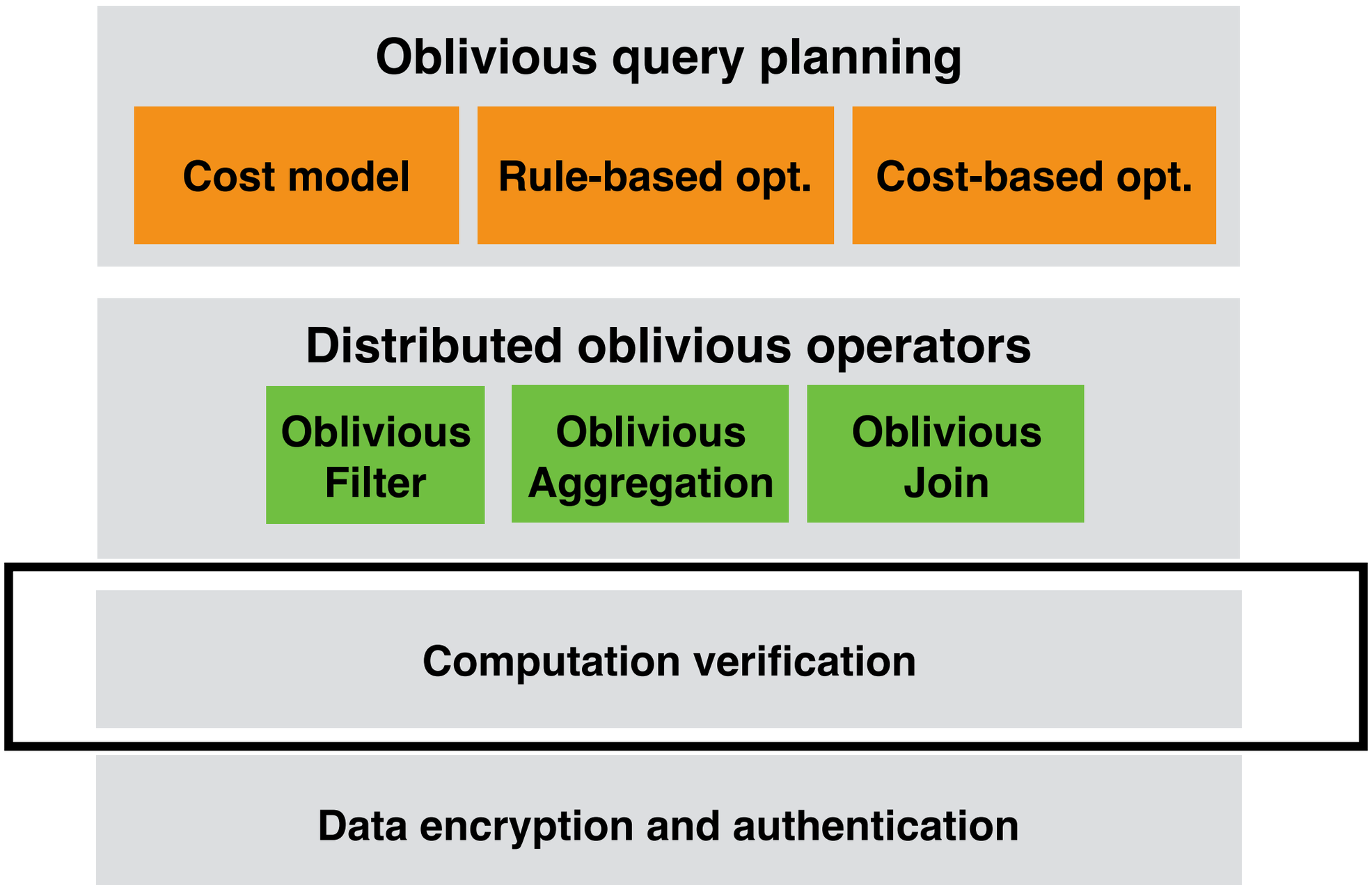
- **Data encryption** and **authentication**
- **Computation integrity:** the client can check that the computation result was not affected by an attacker
- **Obliviousness:** The memory and network accesses of a query is the *same* for *any* two inputs with the same size characteristics (input/outputs)
  - When enabling padding, Opaque hides output sizes as well

Achieving practical obliviousness is not easy

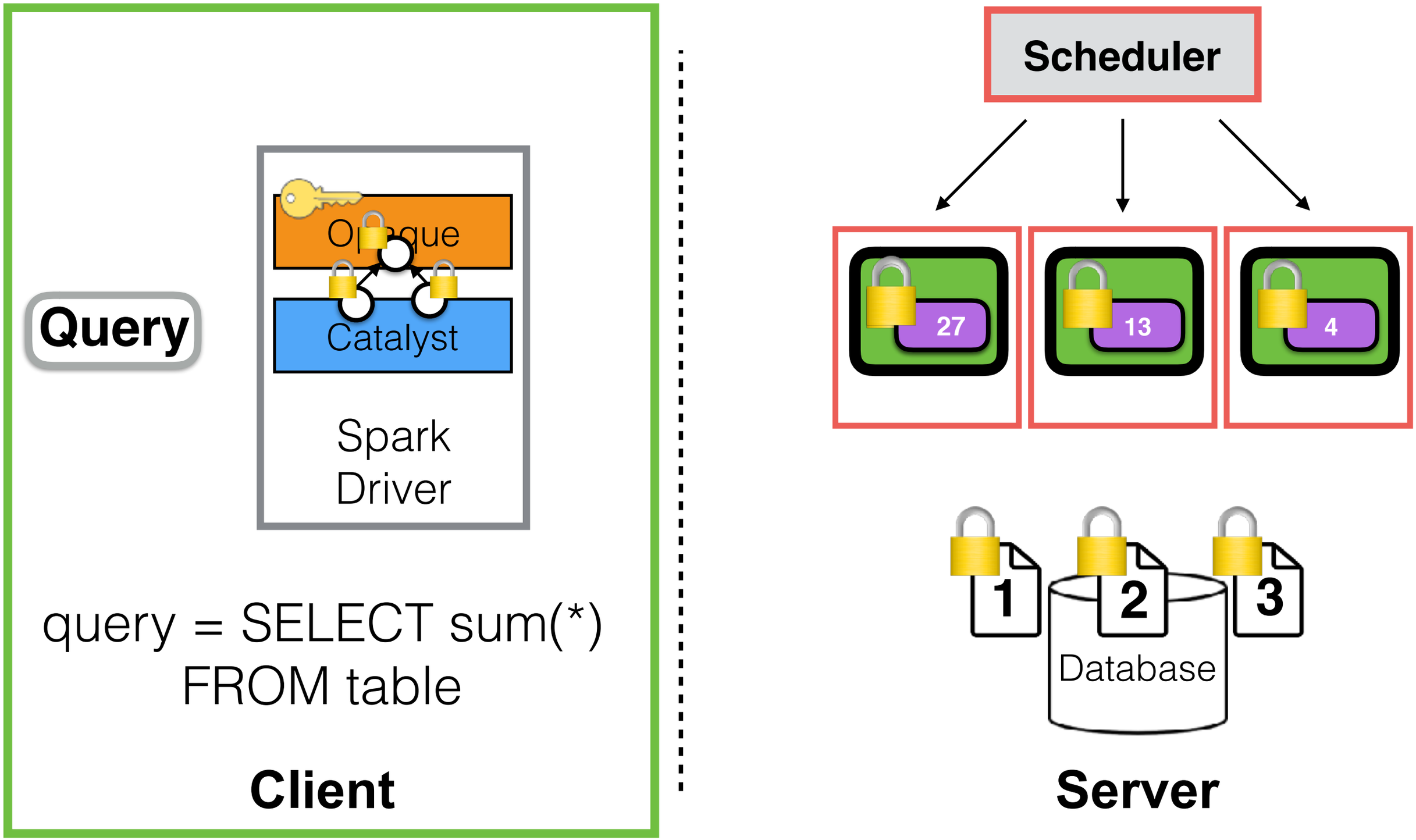
Obliviousness typically comes with high overheads

- For example, the state-of-the-art system, ObliVM, is six orders of magnitude slower than regular computation

Opaque components



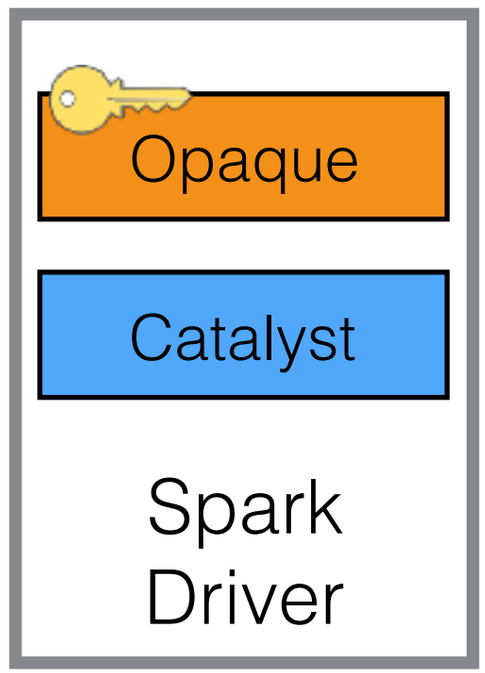
Query execution



Problem: cloud can alter distributed computation

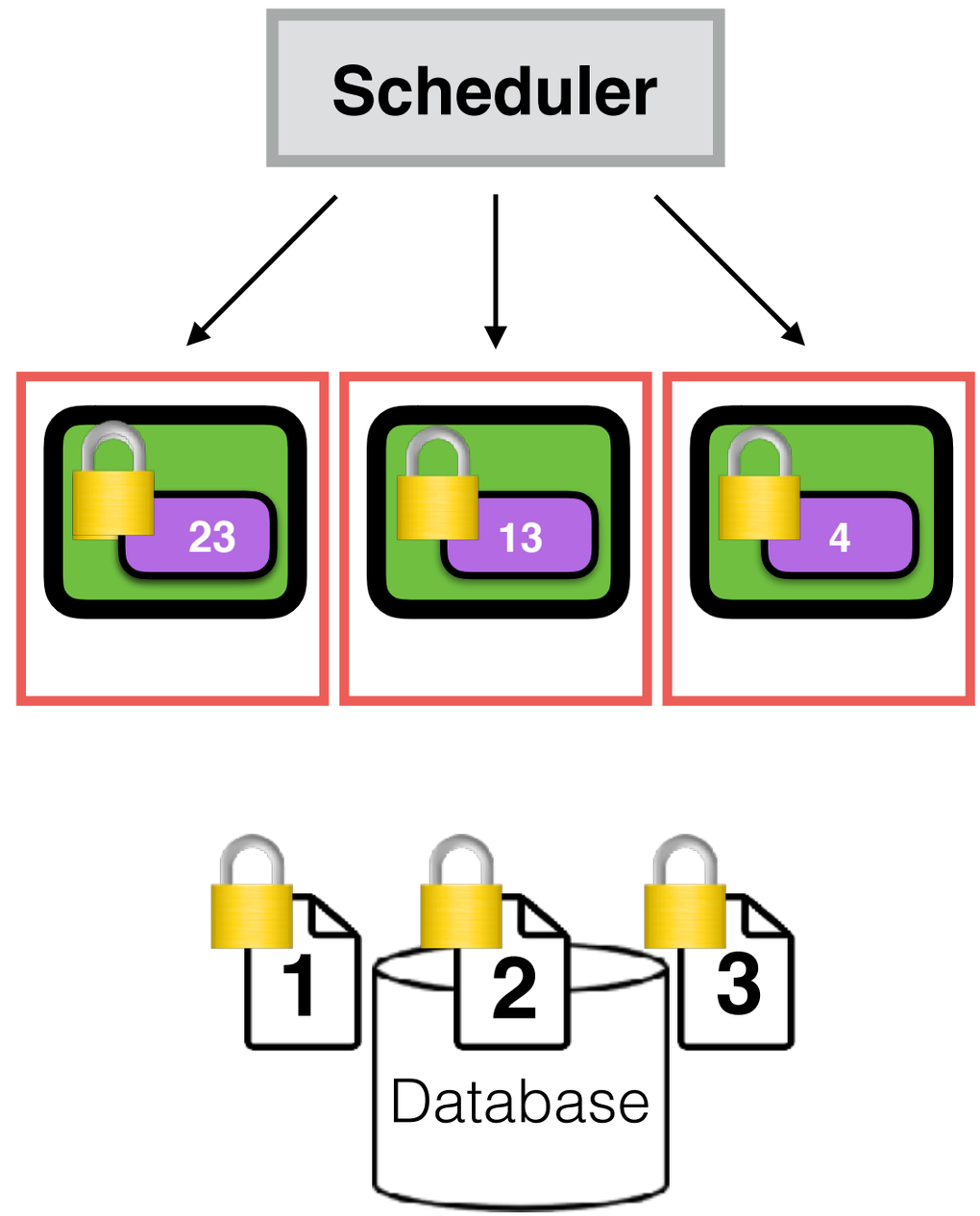
- Drop data
- Modify data
- Skip task
- Replay old state

Example: drop data



query = SELECT sum(*)
FROM table

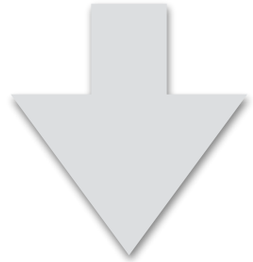
Client



Server

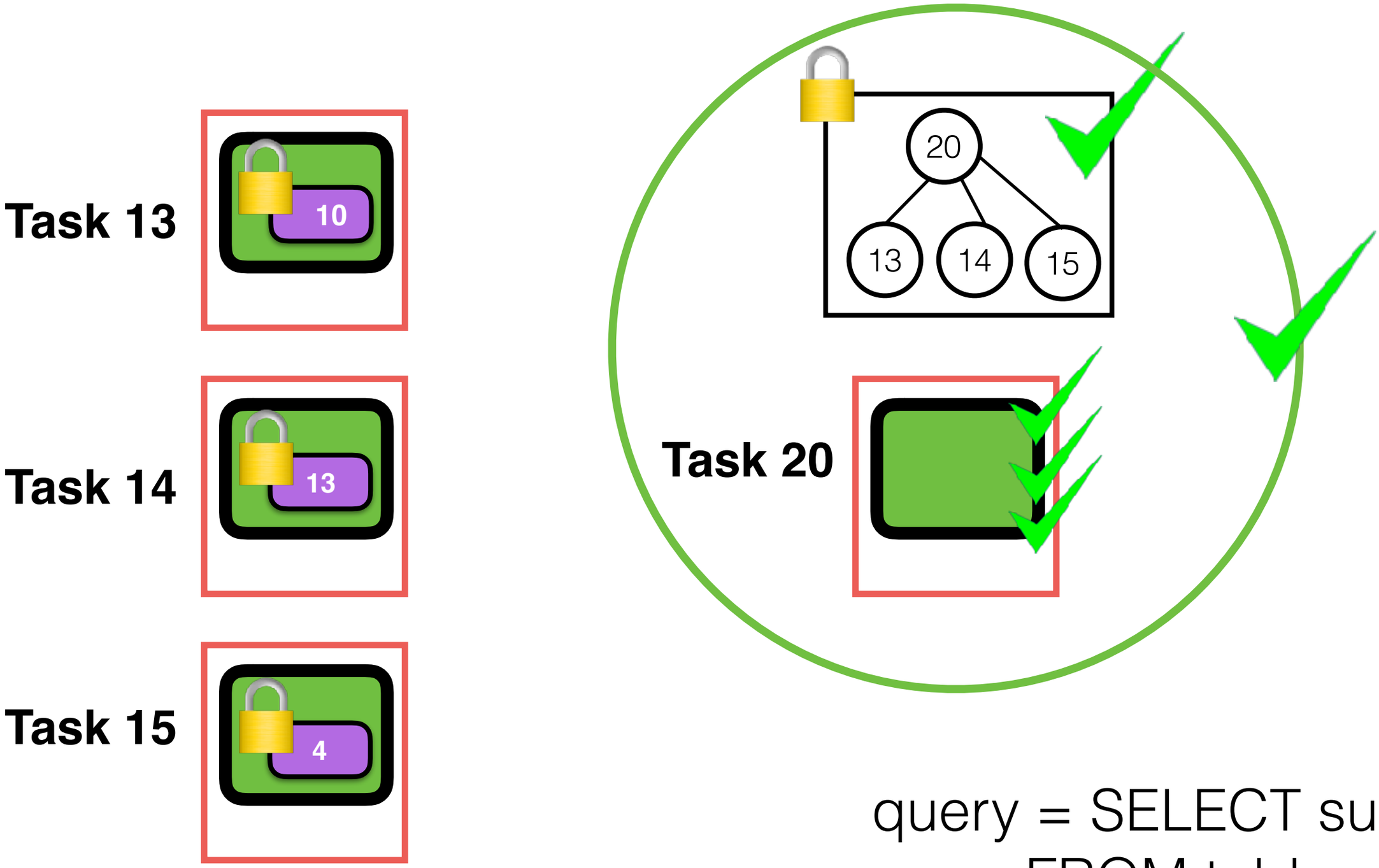
Self-verifying computation

Invariant: if computation does not abort, the execution completed so far is correct



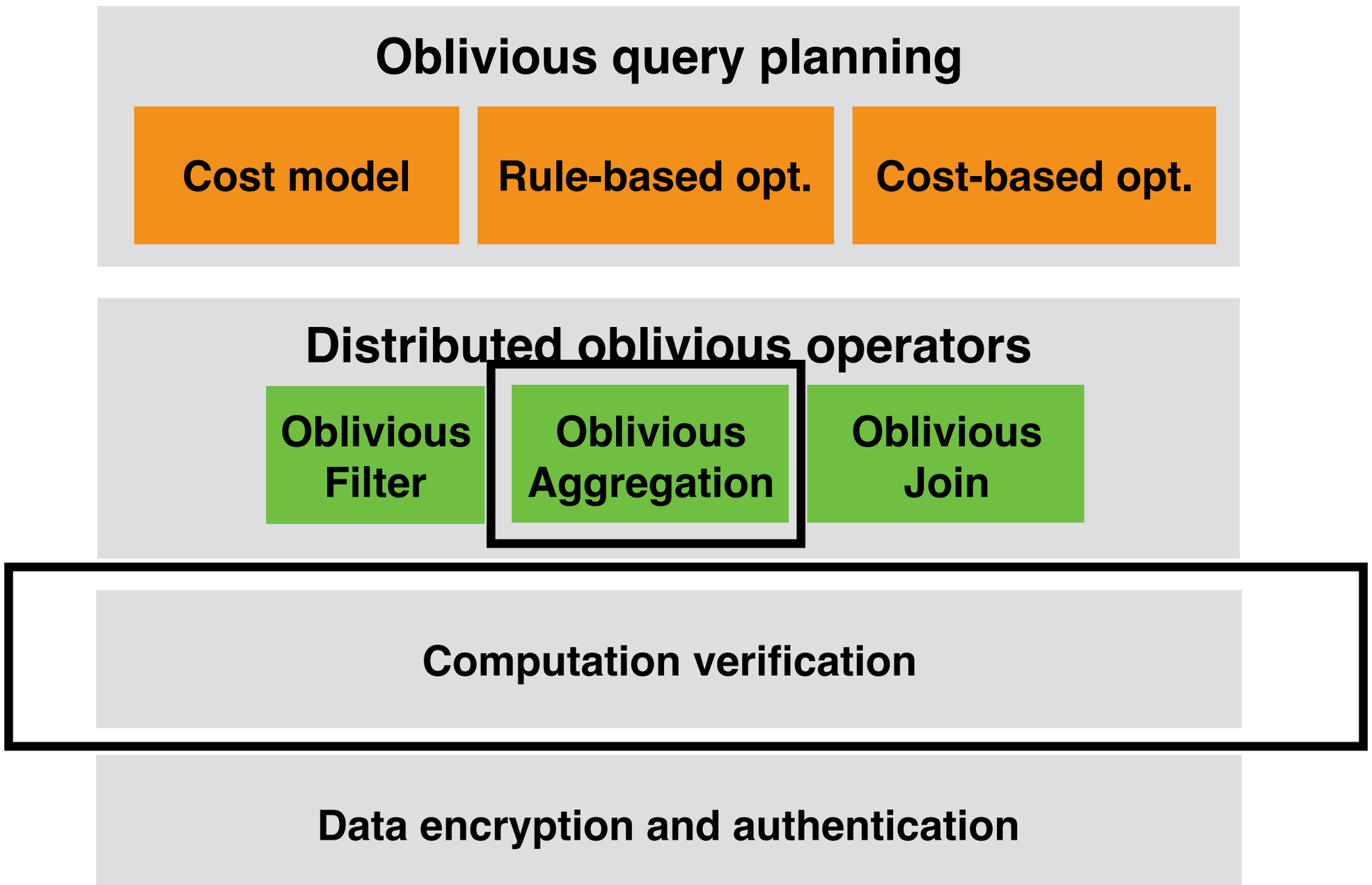
If the computation is complete, then the entire query was executed correctly

Self-verifying computation



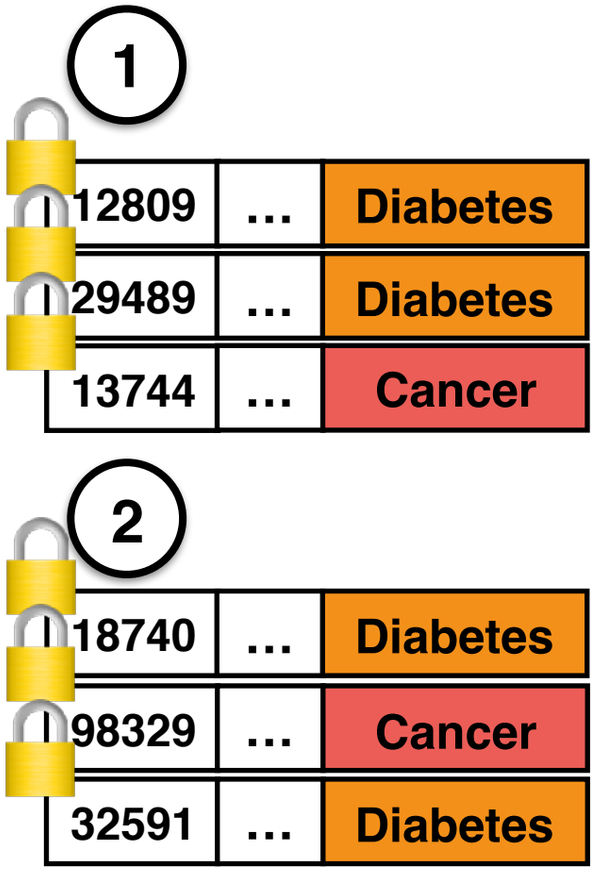
query = SELECT sum(*)
FROM table

Opaque components



Oblivious aggregation

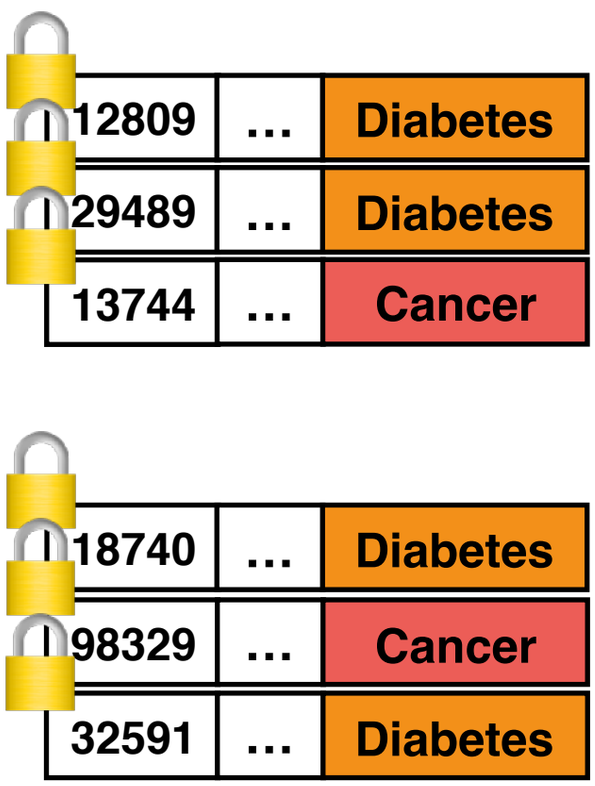
SELECT count(*) FROM medical GROUP BY disease

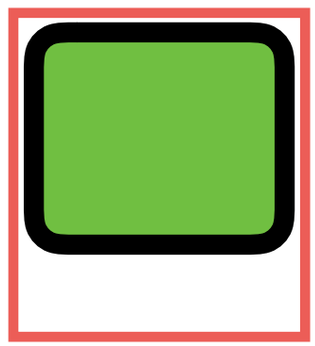


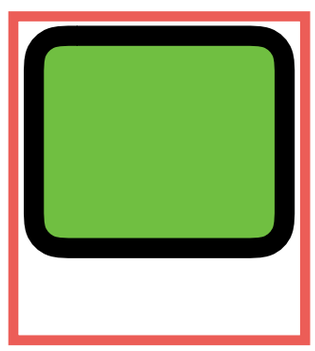
There can be many partitions

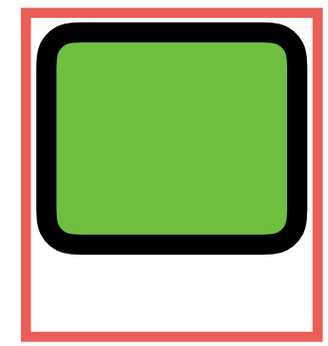
Oblivious aggregation

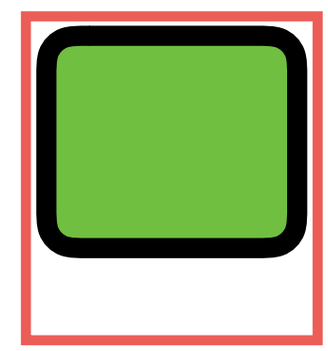
SELECT count(*) FROM medical GROUP BY disease









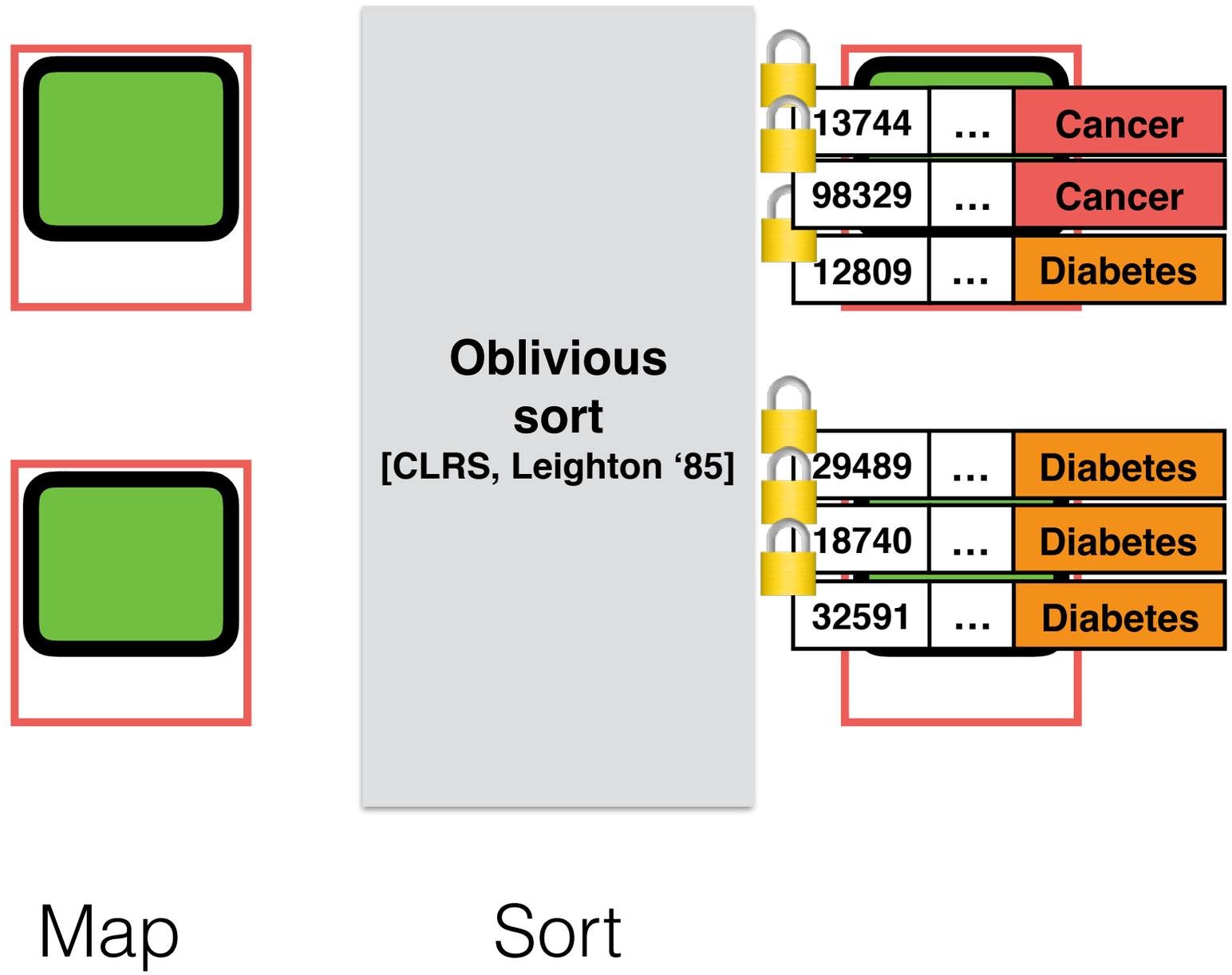


Map

Sort

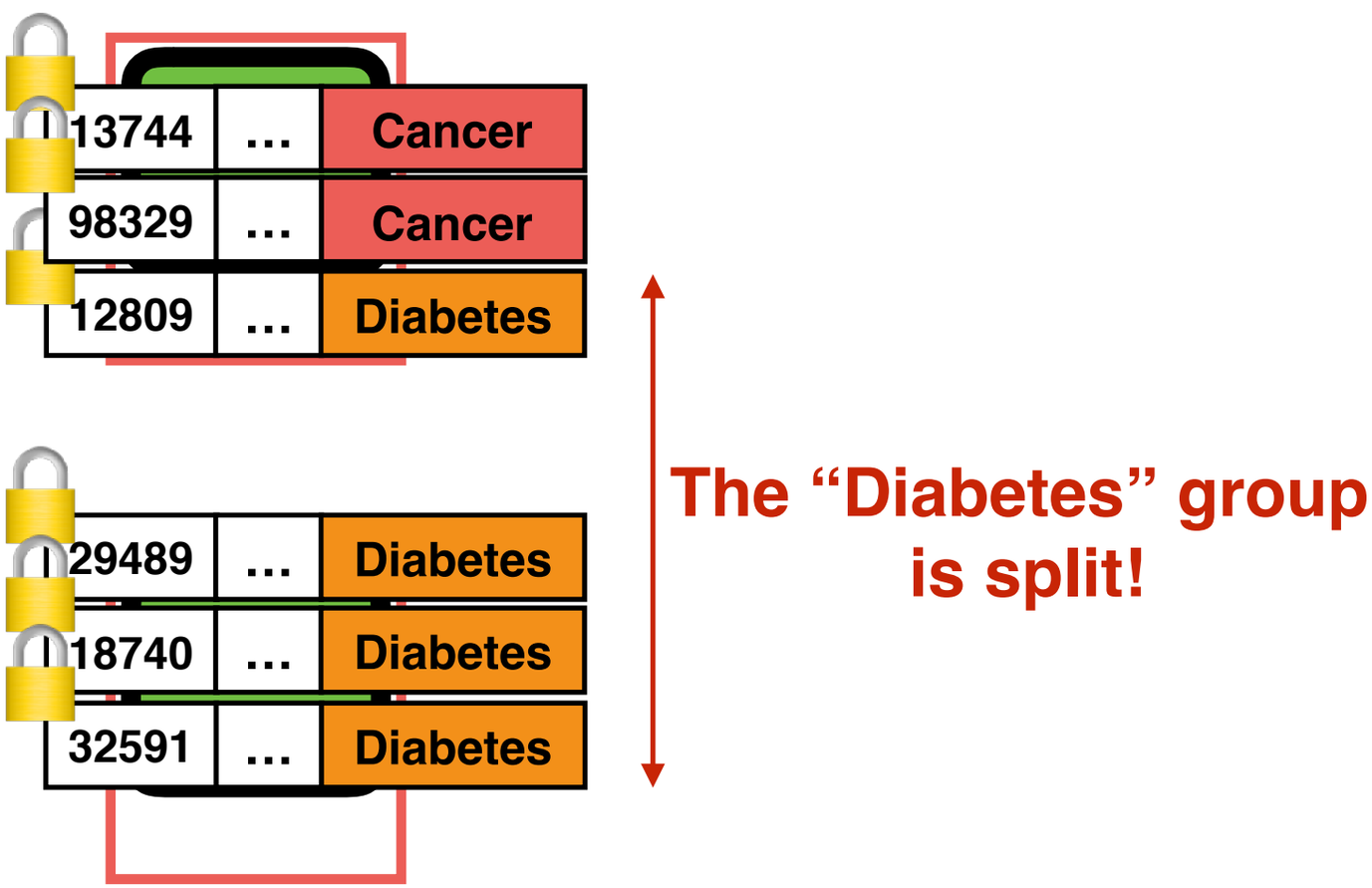
Oblivious aggregation

SELECT count(*) FROM medical GROUP BY disease



Oblivious aggregation

SELECT count(*) FROM medical GROUP BY disease

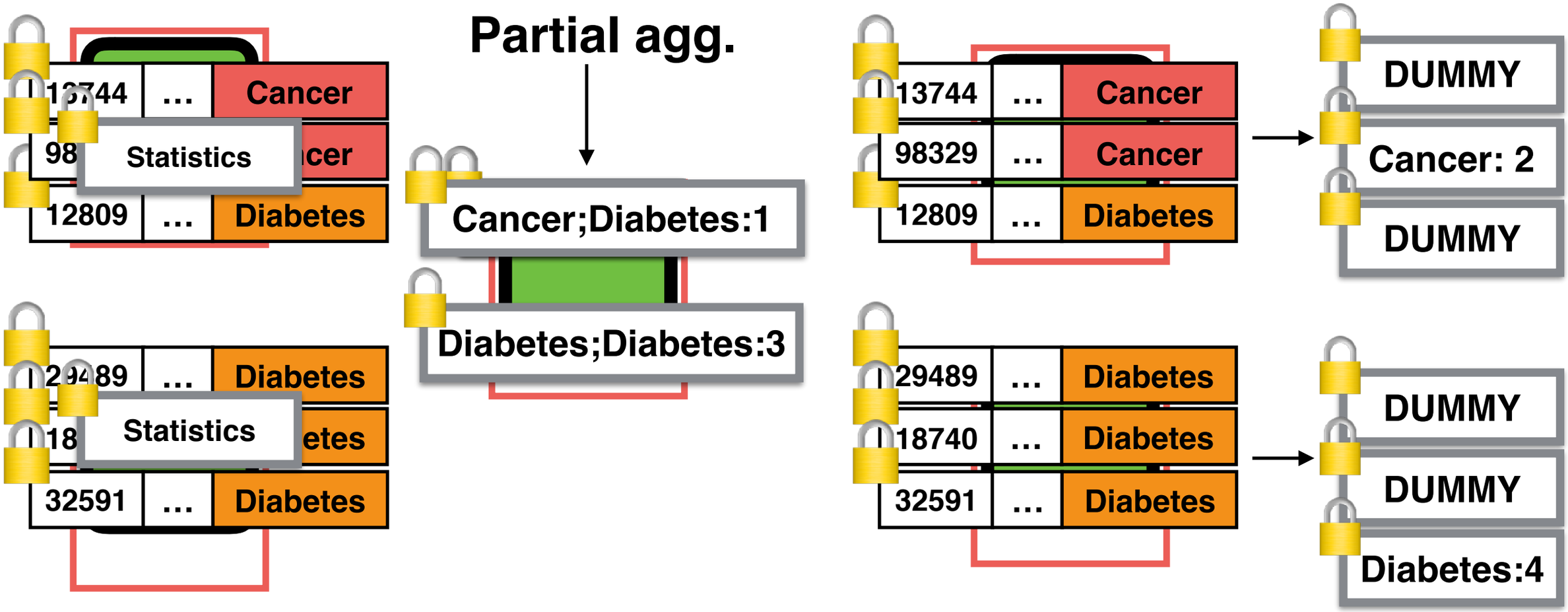


How to aggregate *obliviously* and in *parallel*?

It can span over many partitions

Oblivious aggregation

SELECT count(*) FROM medical GROUP BY disease



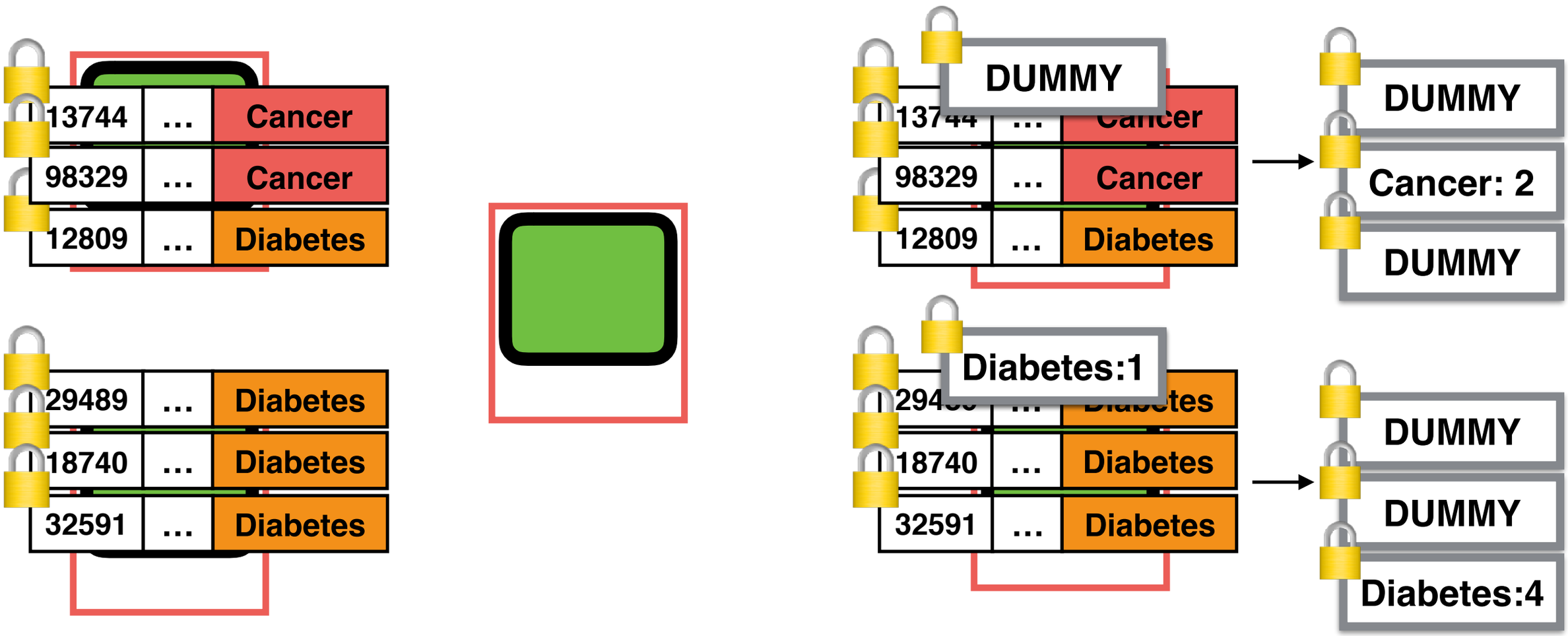
Scan

Boundary processing

Scan

Oblivious aggregation

SELECT count(*) FROM medical GROUP BY disease



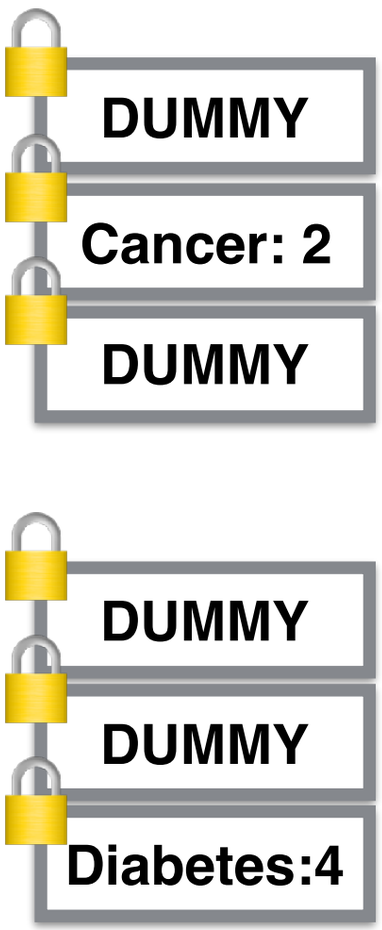
Scan

Boundary processing

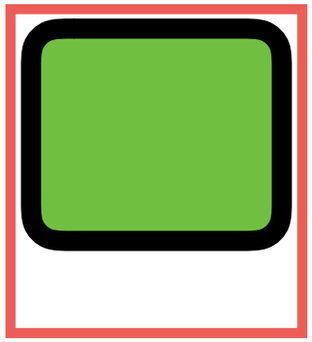
Scan

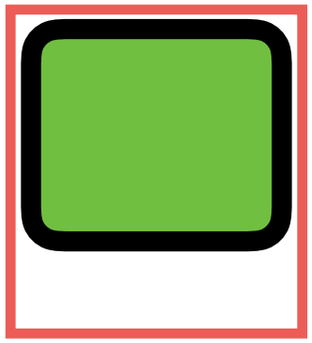
Oblivious aggregation

SELECT count(*) FROM medical GROUP BY disease



Sort





Final result

Aggregation has <u>two</u> sorts…

Can we do better?