

# Practical Security and Privacy for Database Systems

---

SIGMOD 2021 Tutorial

# About the Presenters



**Xi  
He**

*University of  
Waterloo*

**Jennie  
Rogers**

*Northwestern  
University*

**Jholes  
Bater**

*Duke  
University*

**Ashwin  
Machanavajjhala**

*Duke  
University*

**Chenghong  
Wang**

*Duke  
University*

**Xiao  
Wang**

*Northwestern  
University*

*"privacy, databases,  
secure computation,  
machine learning"*

*"privacy-preserving  
analytics, federated  
databases,  
polystores"*

*"privacy-preserving  
analytics, federated  
databases,  
differential privacy"*

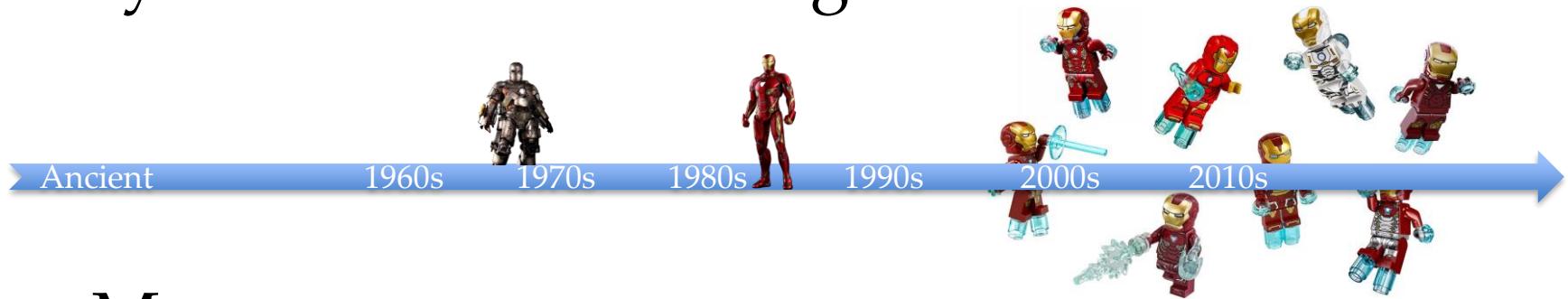
*"privacy, databases,  
differential privacy,  
secure  
computation"*

*"applied  
cryptography;  
differential privacy;  
database security"*

*"multi-party  
computation, zero-  
knowledge  
proof, post-  
quantum  
cryptography"*

# The Success of DBMS

- One of the most important and popular systems for data management



- Many reasons:
  - Logical data model; declarative queries/updates
  - Multi-user concurrent access
  - Safety from system failures
  - Performance, performance, performance
  - ....

# Attacks and Concerns



Riding with the Stars: Passenger Privacy in the NYC

SEPTEMBER 15, 2014 BY ATOCKAR

LEAVE A COMMENT



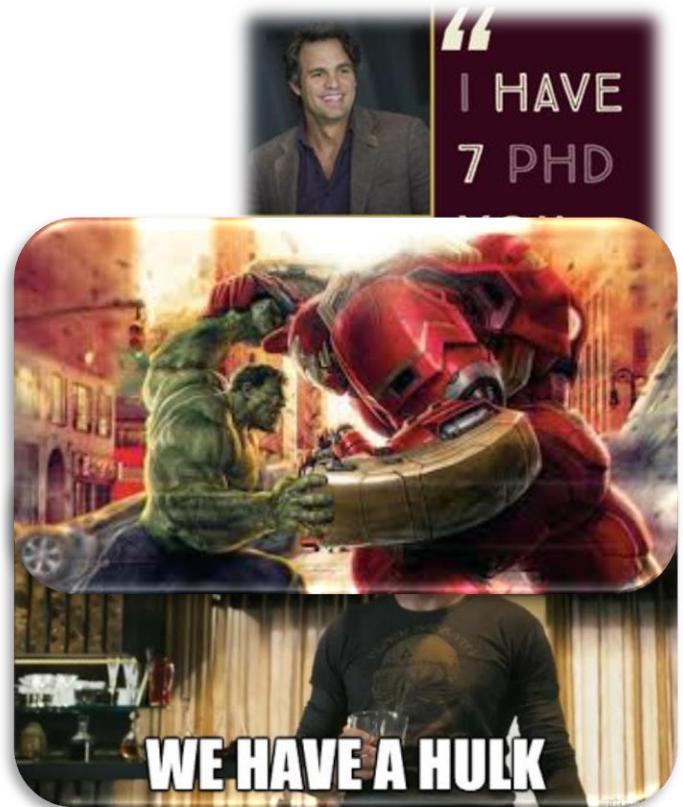
A Face Is Exposed for AOL Searcher No. 4417749

By MICHAEL BARBARO and TOM ZELLER Jr.  
Published: August 9, 2006

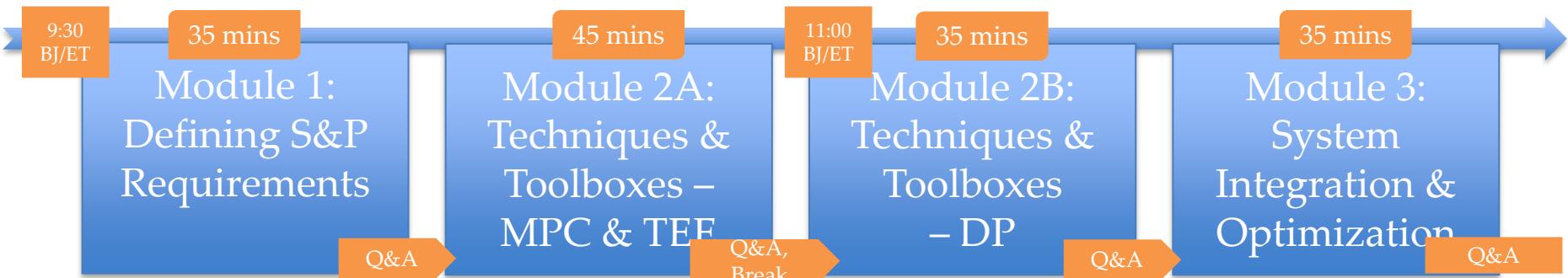


# Let's fight it back, but ...

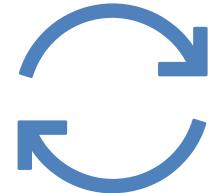
- Experts in security and privacy (S&P) are needed
  - Specialized solutions are not transferable
- Conflict goals
  - Pay a utility/performance cost
  - You don't get the best deal
- S&P is easily breakable
  - Data storage, query processing, output releasing



# Focus of the Tutorial



Principles



Primitives



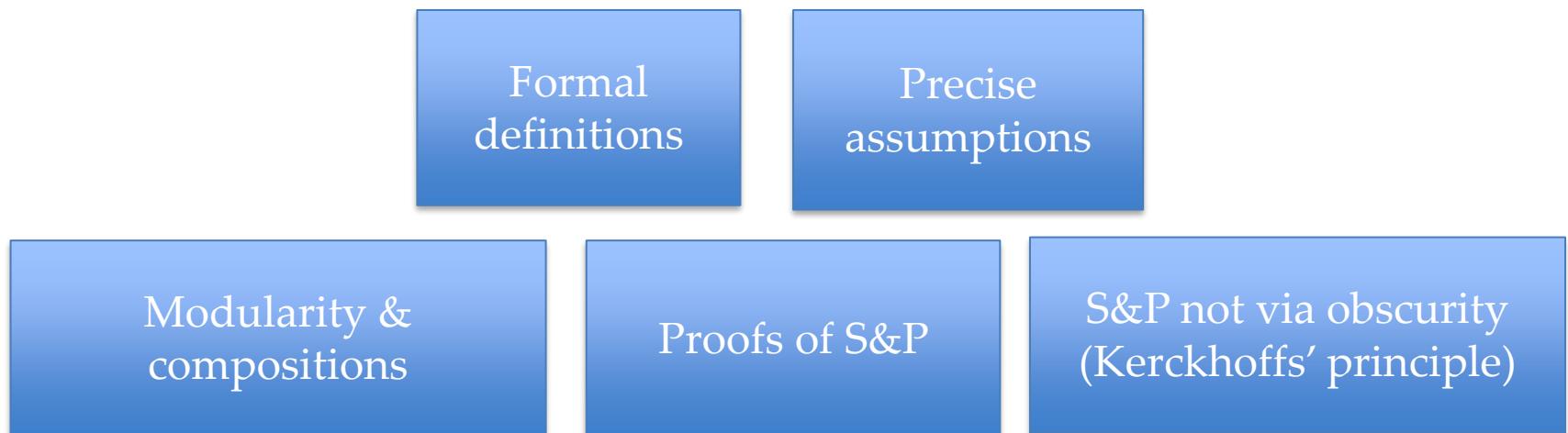
Integration

# **MODULE 1**

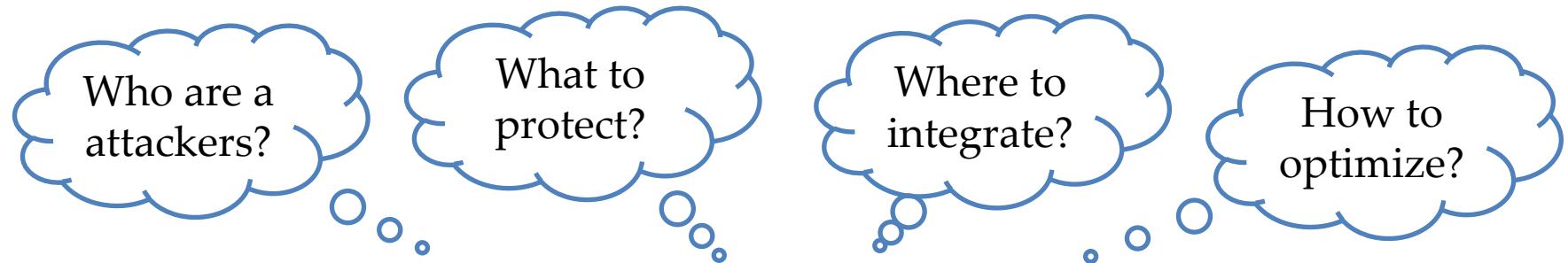
# **DEFINING S&P REQUIREMENTS**

# Provable S&P Requirements

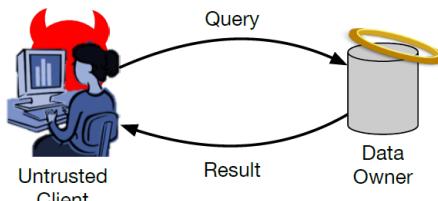
*“...do not end the age-old battle between attacker and defender, but it does provide a framework that helps shift the odds in the defender’s favor.” --- J.K & Y.L*



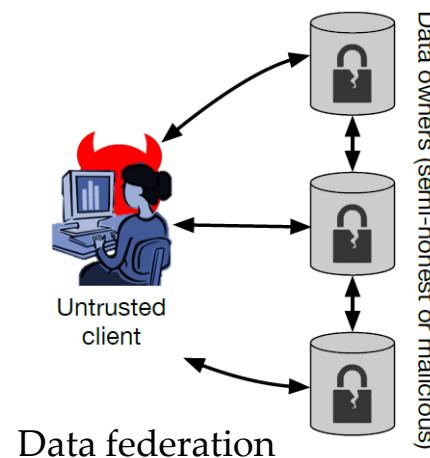
# Defining S&P for DBMS



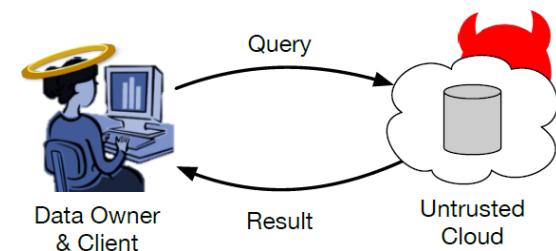
Greatly depend on  
the architecture setup and trust assumptions



Client-server



Data federation



Cloud service provider

# Trust Assumptions

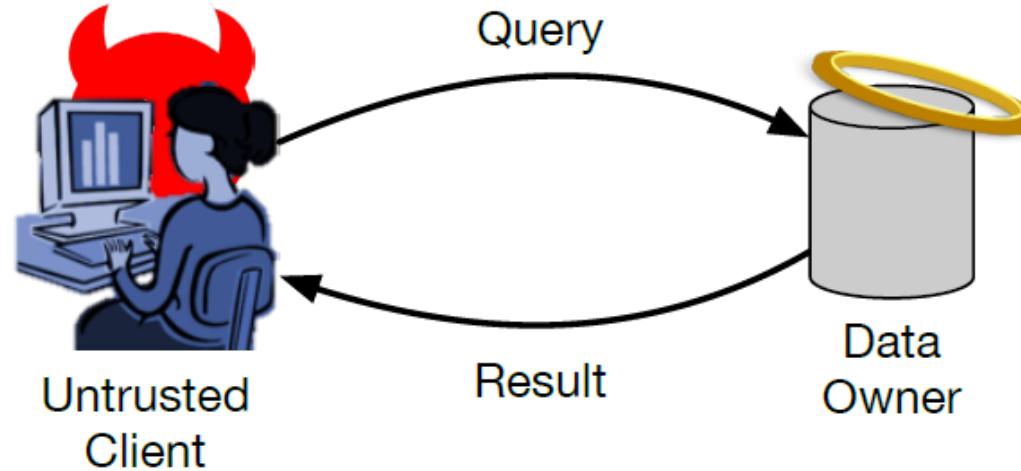
- Malicious party
  - May lie about following protocol
  - Intend to gain unauthorized access/update to private information
- Semi-honest party
  - Follow the protocol faithfully
  - But try to learn everything they can

# Existing Techniques for S&P

Privacy guarantees	Client-server	Data federation	Cloud service provider
Input Data	Differential privacy		N/A
Query Evaluation	N/A	Local DP, Secure multi-party computation, TEE	
Queries	N/A	Private function evaluation	Private information retrieval

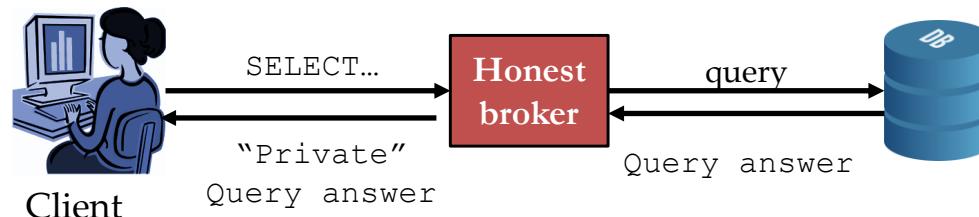
- **Semi-honest party**
  - Follow the protocol faithfully
  - But try to learn everything they can

# Setting #1: Client-Server



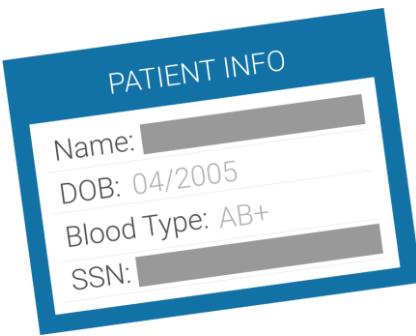
# Client-Server Setting

- Trusted data broker (on behalf of data owners):
  - Have the true and plaintext data stored/processed on a central server according to a valid computation
- Untrusted client (e.g. data analyst)
  - Infer sensitive information about individuals from the released output by the data broker/ data owner  
→ “Data Privacy”

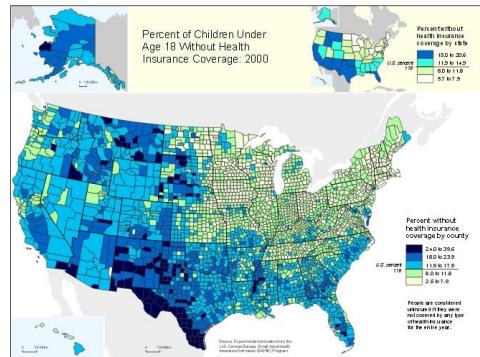


# Conventional Privatization Method

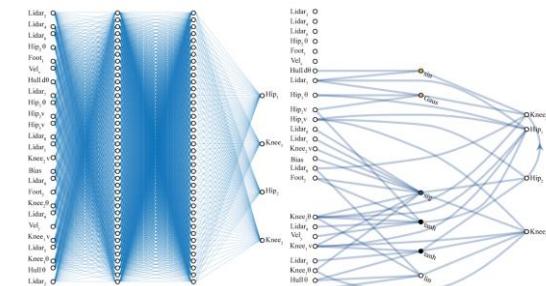
## De-identified records (e.g., medical)



# Statistics (e.g., demographic)



## Predictive models (e.g., advertising)



# What could possibly go wrong?

**A Face Is Exposed for AOL Searcher No. 4417749**

By MICHAEL BARBARO and TOM ZELLER Jr.  
Published: August 9, 2006



**Wherefore Art Thou R3579X? Anonymized Social Networks, Hidden Patterns, and Structural Steganography**

Lars Backstrom  
Dept. of Computer Science  
Cornell University, Ithaca NY  
lars@cs.cornell.edu

Cynthia Dwork  
Microsoft Research  
dwork@microsoft.com

Jon Kleinberg  
Dept. of Computer Science  
Cornell University, Ithaca NY  
kleinber@cs.cornell.edu

**PATIENT INFO**

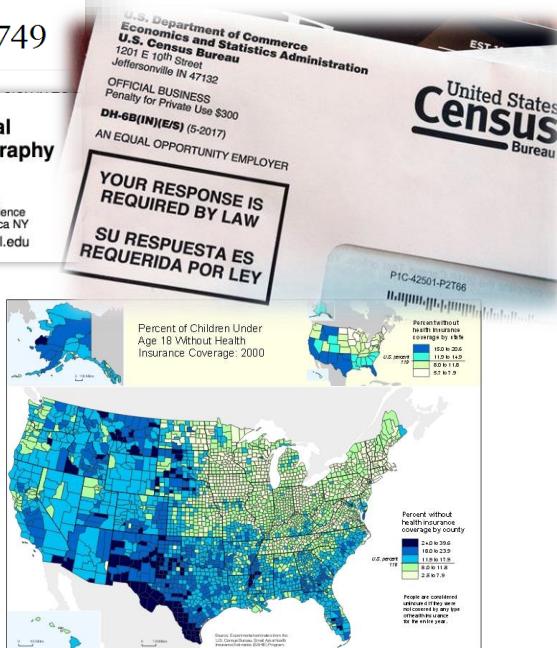
Name: [REDACTED]  
DOB: 04/2005  
Blood Type: AB+  
SSN: [REDACTED]

**U.S. Department of Commerce  
Economics and Statistics Administration  
U.S. Census Bureau  
Jefferson Street  
1201 E 10th Street  
Jeffersonville IN 47132**

**OFFICIAL BUSINESS**  
Penalty for Private Use \$300  
**DH-6B(IN)(B) (5-2017)**

**AN EQUAL OPPORTUNITY EMPLOYER**

**YOUR RESPONSE IS REQUIRED BY LAW**  
**SU RESPUESTA ES REQUERIDA POR LEY**



**Membership Inference Attacks Against Machine Learning Models**

Reza Shokri  
Cornell Tech

Marco Stronati\*  
INRIA

Congzheng Song  
Cornell

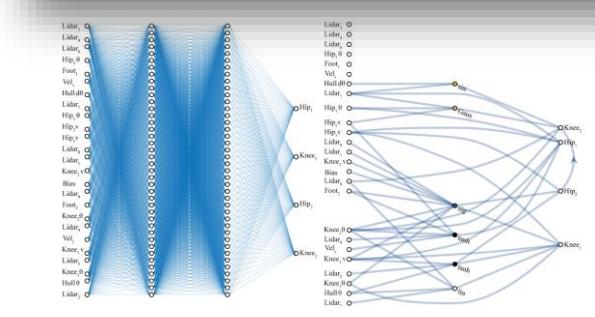
Vitaly Shmatikov  
Cornell Tech

**Membership Inference Attack on Graph Neural Networks**

Iyiola E. Olatunji  
L3S Research Center,  
Hannover, Germany.  
iyiola@l3s.de

Wolfgang Nejdl  
L3S Research Center,  
Hannover, Germany.  
nejdl@l3s.de

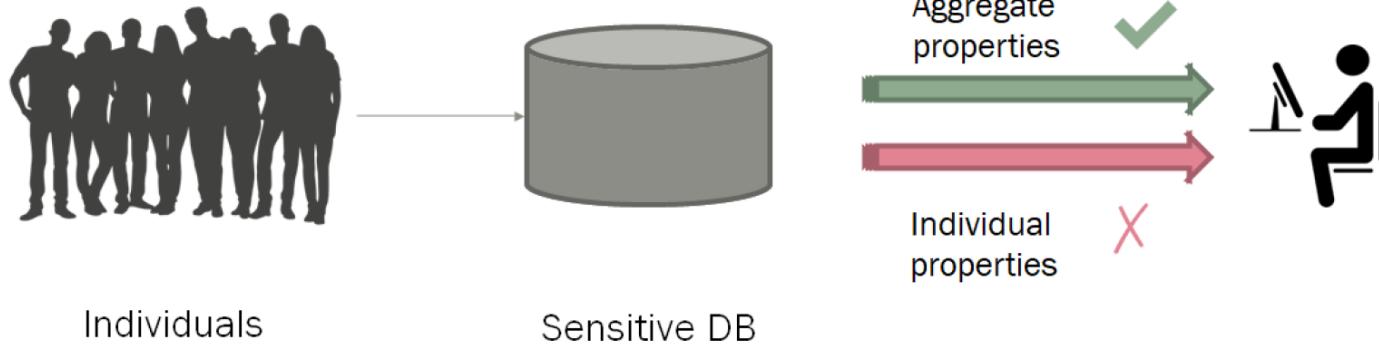
[Arxiv21]  
Megha Khosla  
L3S Research Center,  
Hannover, Germany.  
khosla@l3s.de



**Fundamental Law of Info Reconstruction [DN03]**  
 “overly accurate” estimates of “too many” statistics is blatantly non-private.

# Techniques for S&P

Privacy Guarantees	Client-server	Data federation	Cloud service provider
Input Data	Differential privacy		N/A
Query Evaluation	N/A	Local DP, Secure multi-party computation, TEE	
Queries	N/A	Private function evaluation	Private information retrieval



# Differential Privacy [D06]

- Goal: Protect the privacy of individuals in the database while releasing the output of a valid computation to untrusted client



[Machanavajjhala et al, ICDE 2008]

[uber-archive/sql-differential-privacy](#)



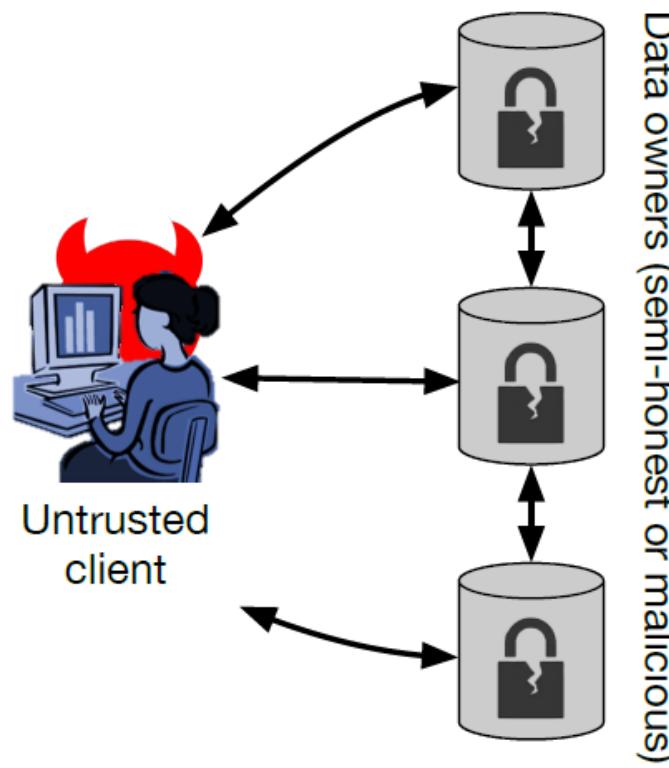
Dataflow analysis & differential privacy for SQL queries. This project is deprecated and not maintained.

5 Contributors    5 Issues    362 Stars    67 Forks

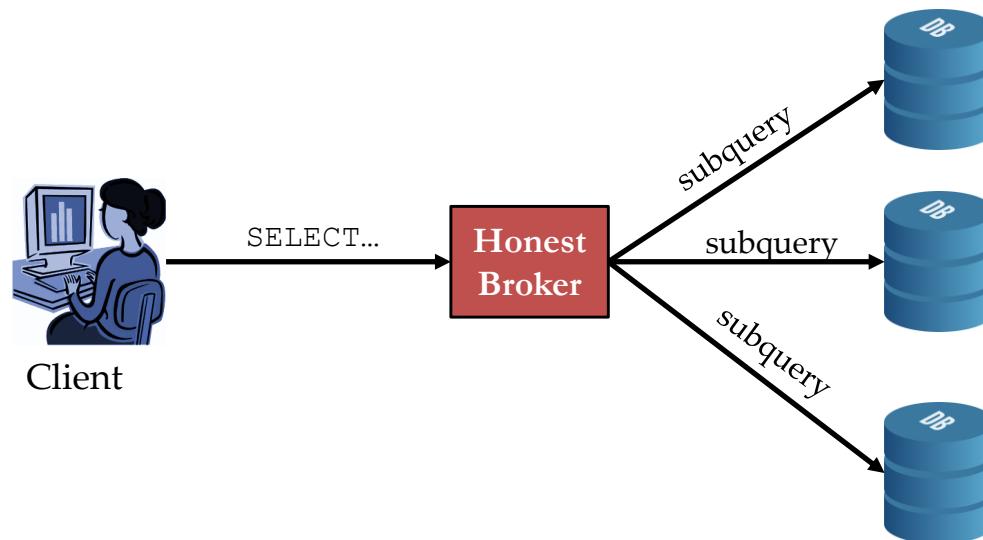
[Johnson et al, VLDB 2018]

- A provable privacy guarantee
- Trade-offs: accuracy and privacy

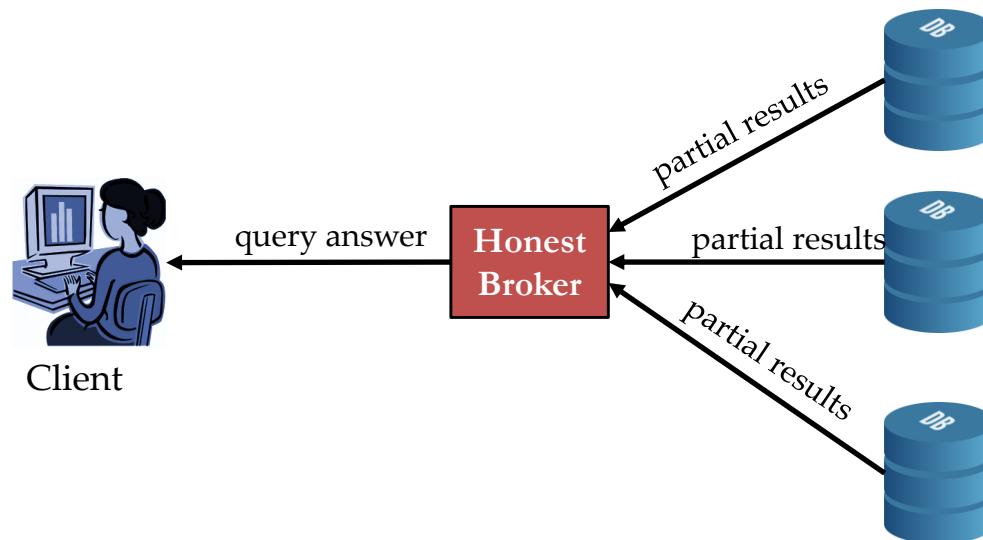
# Setting #2: Data Federation



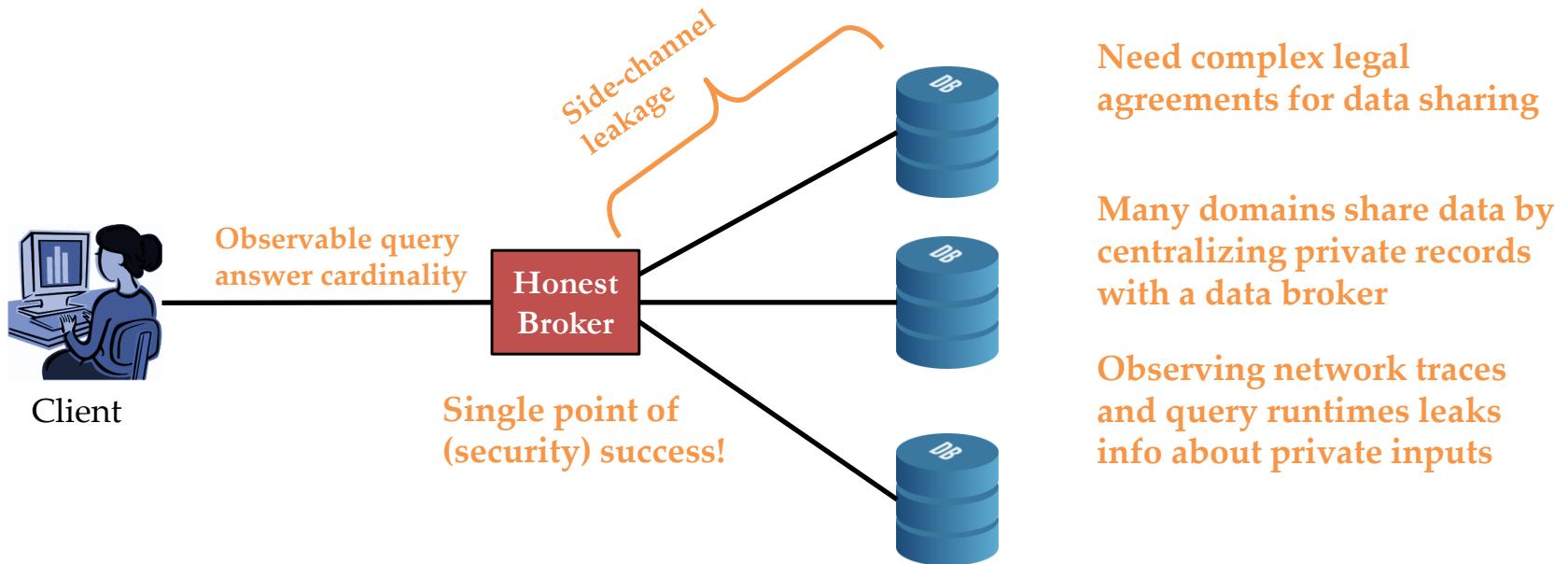
# Conventional Data Federation



# Conventional Data Federation



# What could possibly go wrong?



# Conventional Data Federation

```
SELECT COUNT(DISTINCT patient_id)  
FROM diagnosis  
WHERE diagnosis_code='covid';
```



Client

Untrusted | Secure

Honest  
Broker



# Private Data Federation

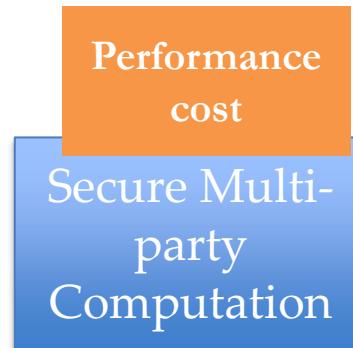
- Untrusted client/broker (e.g. data analyst)
  - “Data Privacy”
- Semi-honest servers
  - Honestly evaluate query over federated data
  - Curious about other’s input data and infer them via the computation (e.g, encrypted data, intermediate results, side channel information)
    - “Computing with Confidentiality”

# Computing with Confidentiality

- Goal: protect confidentiality of data *while* we compute queries over it in an untrusted setting



[Erlingsson et al, CCS 2014]



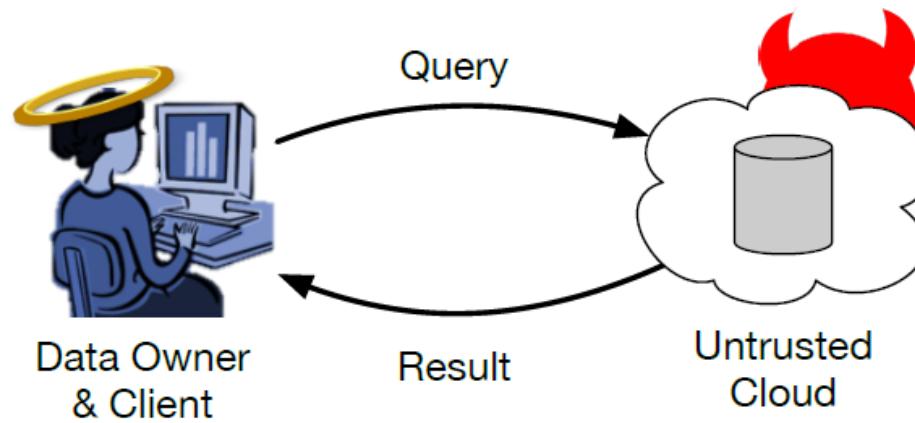
Boston wage gap 2017

- Provable privacy guarantees

# Techniques for S&P

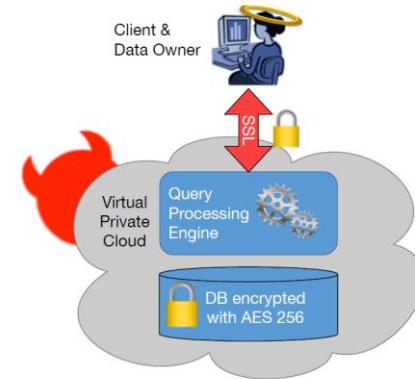
Privacy Guarantees	Client-server	Data federation	Cloud service provider
Input Data	Differential privacy		N/A
Query Evaluation	N/A	Local DP, Secure multi-party computation, TEE	
Queries	N/A	Private function evaluation	Private information retrieval

# Setting #3: Cloud Service Provider



# Conventional Cloud

- Untrusted servers
  - “Computing with confidentiality”
- Trusted client
  - Data owner and data analyst are the same party
  - Allow more information to be returned from the cloud to the client
  - “Private information retrieval”



# Private Information Retrieval<sub>[CGKS95]</sub>

- Goal: Client can retrieve an item from a server in possession of a database without revealing which item is retrieved
- Return entire DB: only protocol for information theoretical privacy in a single server setting [BB15]

Computationally  
bounded server  
[KO97]

Multiple non-  
colluding servers  
[DGH 2012]

- Provable privacy guarantees
- Trade-offs: performance and privacy

# Existing Techniques for S&P

Privacy guarantees	Client-server	Data federation	Cloud service provider
Input Data	Differential privacy		N/A
Query Evaluation	N/A	Local DP, Secure multi-party computation, TEE	
Queries	N/A	Private function evaluation	Private information retrieval

- **Semi-honest party**
  - Follow the protocol faithfully
  - But try to learn everything they can

# Additional Settings & Techniques

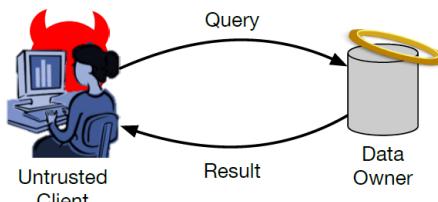
- **Malicious party**
  - Act maliciously
    - Unauthorized updates to the data storage
    - Incorrect query evaluation

Integrity	Client-server	Data federation	Cloud service provider
Storage	Authenticated data structures (ADS): e.g. Merkle tree, blockchain		
Query Evaluation	Zero-knowledge proofs (MPC), TEE, ADS		

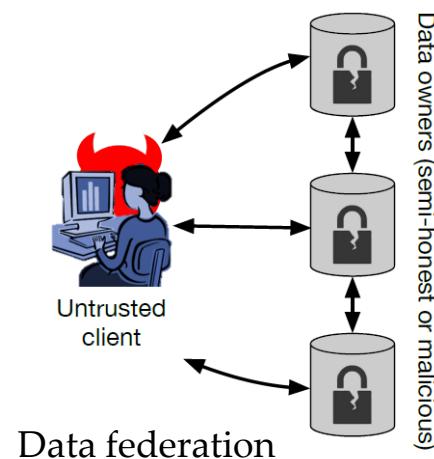
Trade-off: performance and integrity

# Summary

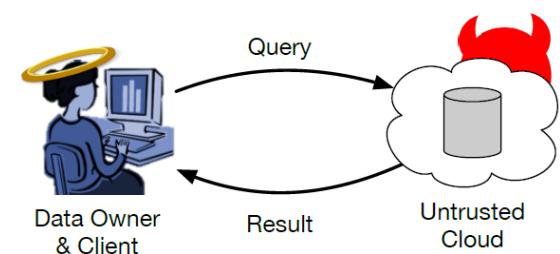
Privacy	Client-server	Data federation	Cloud service provider
Input Data	Differential privacy		N/A
Query Evaluation	N/A	Local DP, Secure multi-party computation, TEE	
Queries	N/A	Private function evaluation	Private information retrieval



Client-server



Data federation



Cloud service provider

# **MODULE 2A**

## **MPC & TEE**

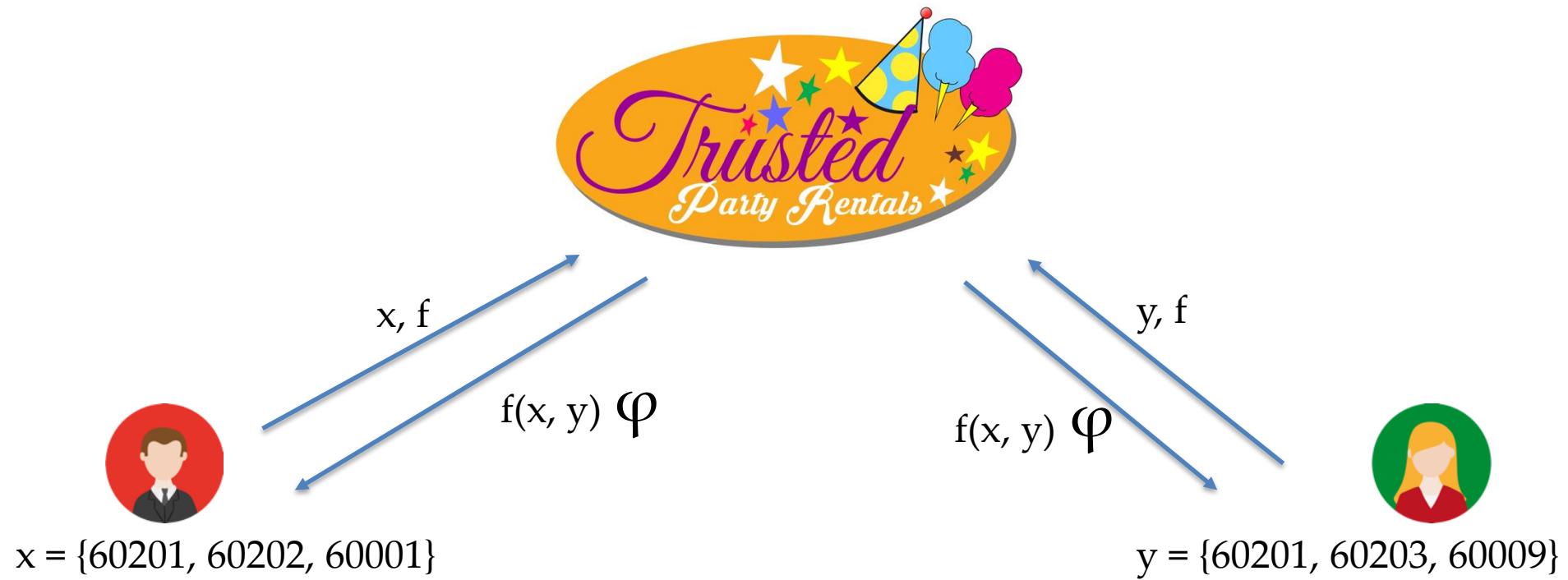
# Overview

- Goal: protect confidentiality of data *while* we compute queries over it in an untrusted setting
- Attack vectors:
  - Data protection: Encrypt data and intermediate results
  - Side channel: Prevent leakage from the query's instruction traces and others

# Computing with Confidentiality

- Why confidentiality is difficult? No one can be trusted:
  - Intend, capability, etc
- Technologies for confidentiality computation bring trustiness

# CC with a Trusted Party



$\varphi$ : Side information

$f(x, y) = \text{intersection of } x \text{ and } y$

# Side Channel Information

- Program trace
- Data access trace

All leaks information!  
Some more less harmful

- Program execution time
- Intermediate result size
- ...

Often need to incur the  
worst-case cost

# DATA AND TRACE OBLIVIOUSNESS

# Program Trace

```
x = {60201, 60202, 60001}  
y = {60201, 60203, 60009}
```

Program trace depends  
on the input!

```
res = 0  
for i in x:  
    for j in y:  
        if i == j:  
            res += 1  
        break;
```

# Data Access Trace

```
def binary_search (val, s, t) :  
    mid = (s + t) / 2;  
    if (val < mem[mid] )  
        bs(val, 0, mid)  
    else  
        bs(val, mid+1, t)
```

Data access trace  
depends on the input!

# Definition

We say a program  $P$  is oblivious if there is an efficient algorithm  $S$ , such that for any input  $I$  to the program,

$\text{Trace}(P, I)$     *is indistinguishable from*     $S(P)$

# An Oblivious Algorithm Example

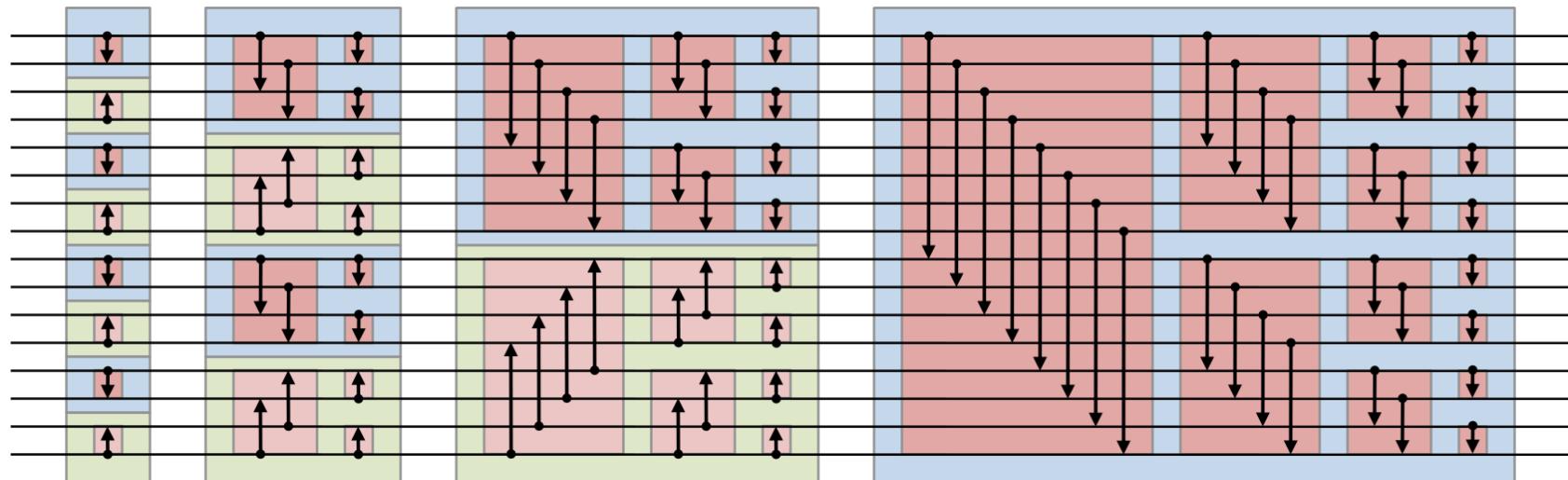
$z = \{60001, 60201, 60201, 60002, 60203, 60009\}$

$z = x \mid\mid y$

res=0

**Trace-independent Sort z**

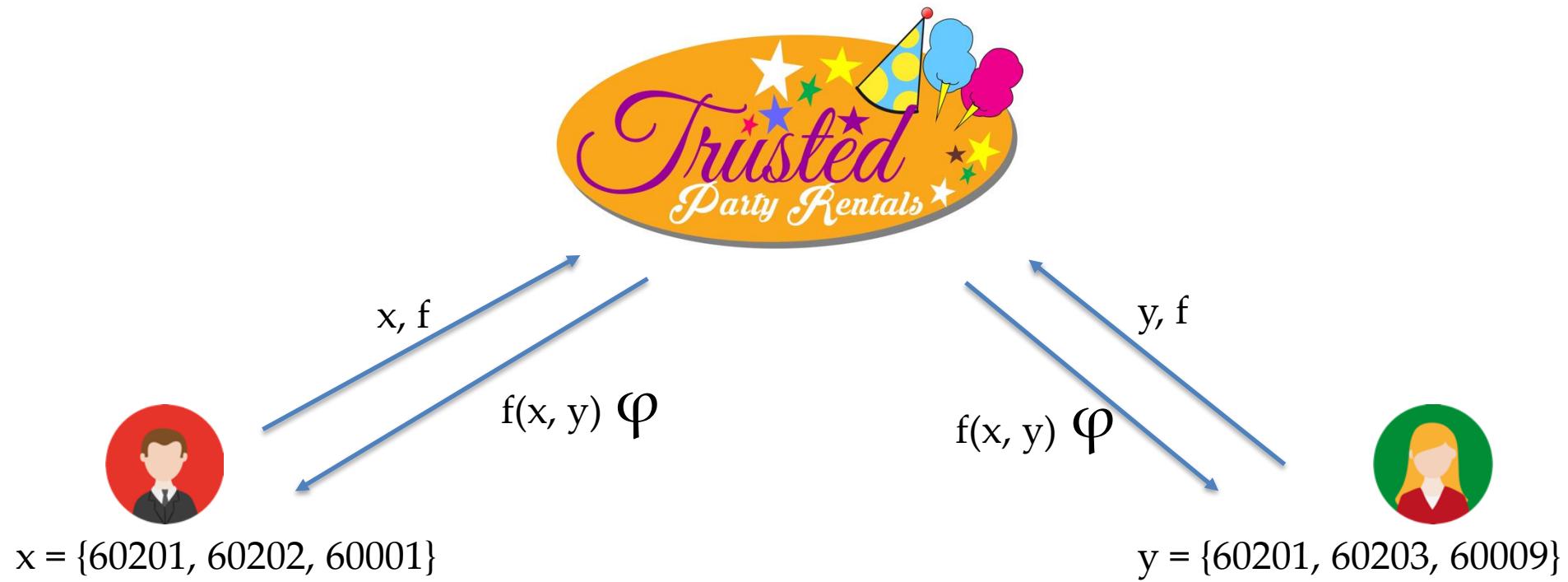
```
for i in [0, len(z)-1]:
    if z[i] == z[i+1]
        res+=1
```



# Recent Works

- Oblivious algorithms
  - Graph-based computation
  - Specific data structures
  - Parallelism
  - More efficient sorting
- Oblivious RAM

# CC with a Trusted Party



## $\varphi$ : leakage profile

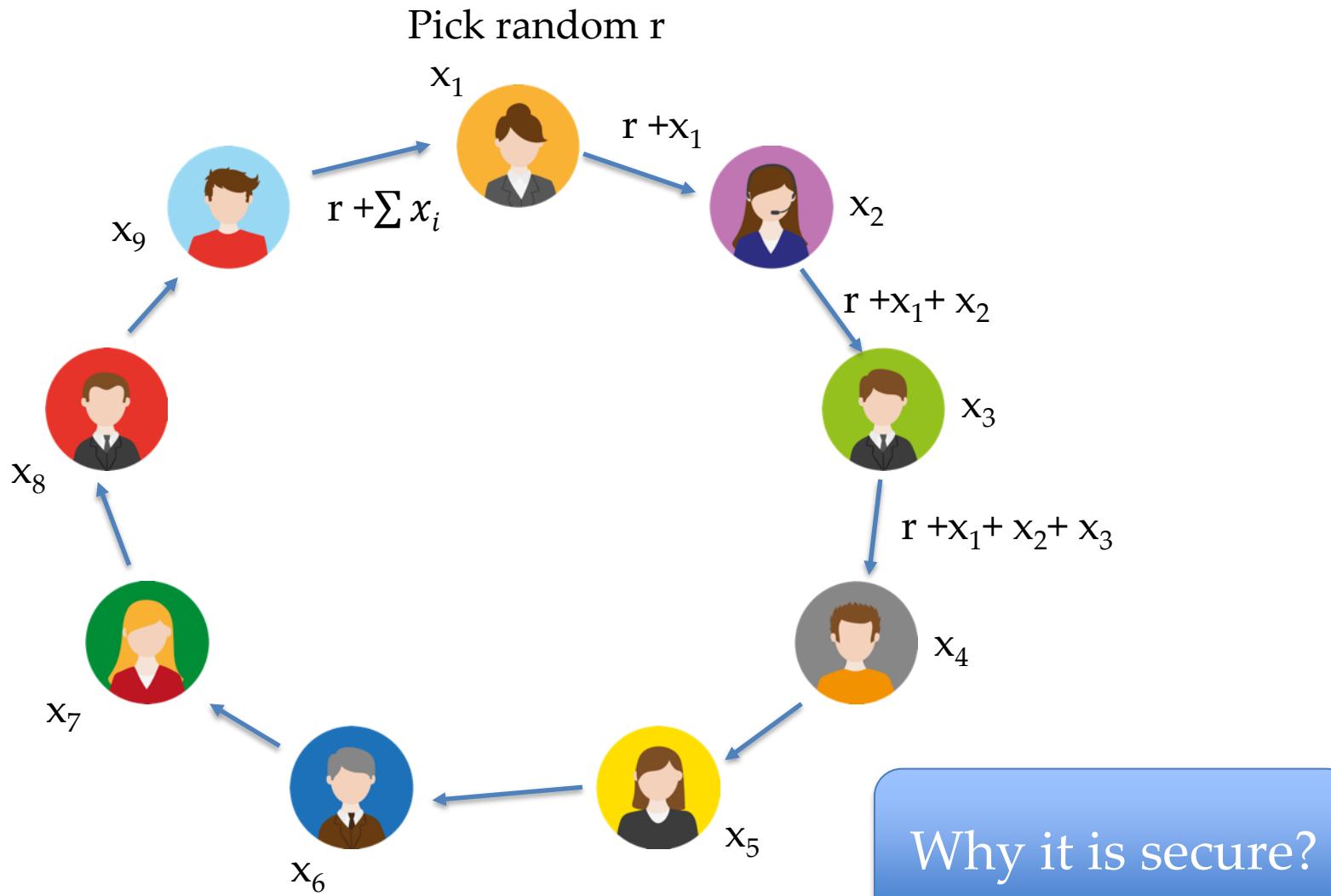
- Program trace
- Data access trace
- Program execution time
- Intermediate result size

...

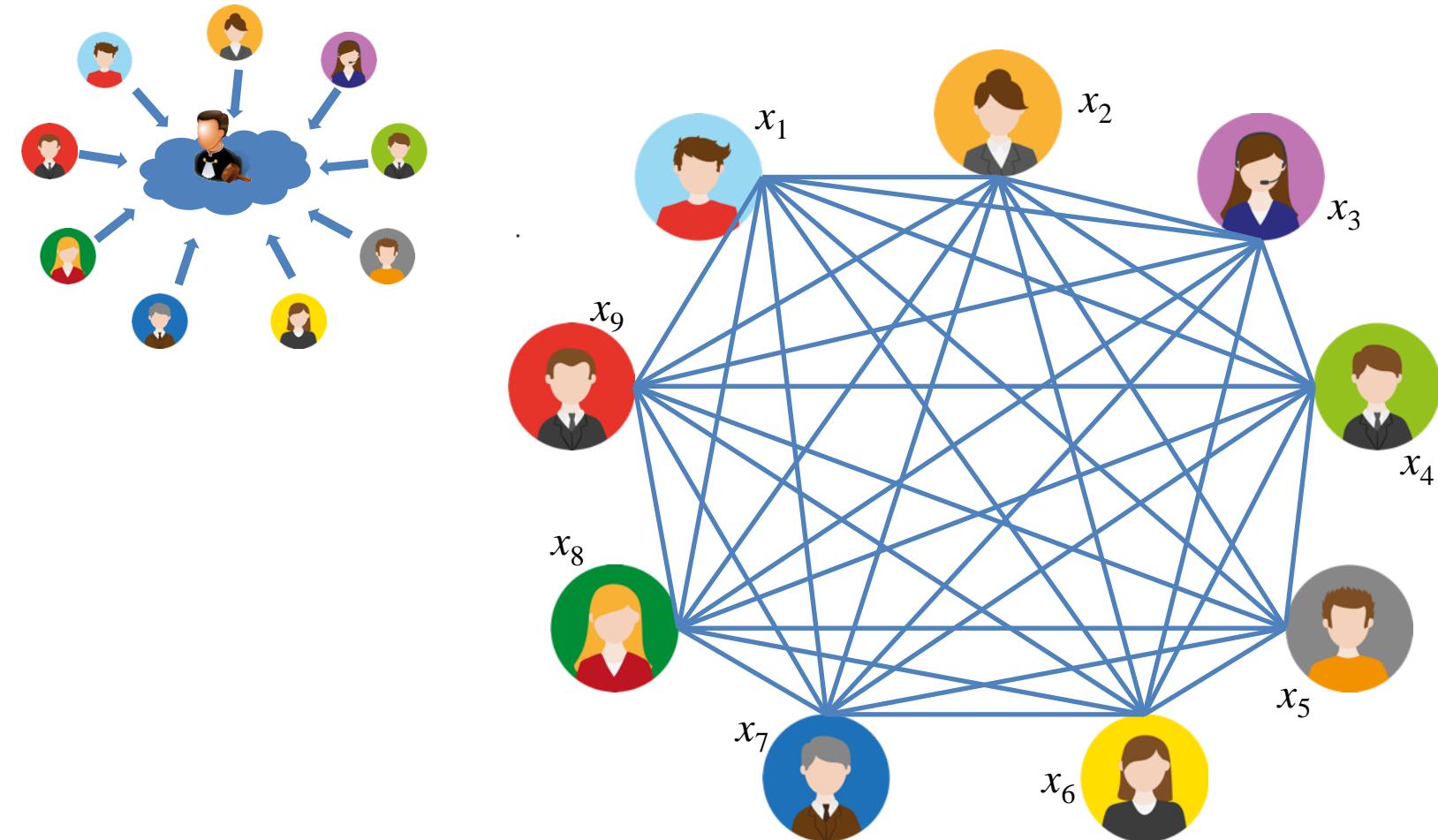
$f(x, y)$  = intersection of  $x$  and  $y$

# **SECURE MULTI-PARTY COMPUTATION**

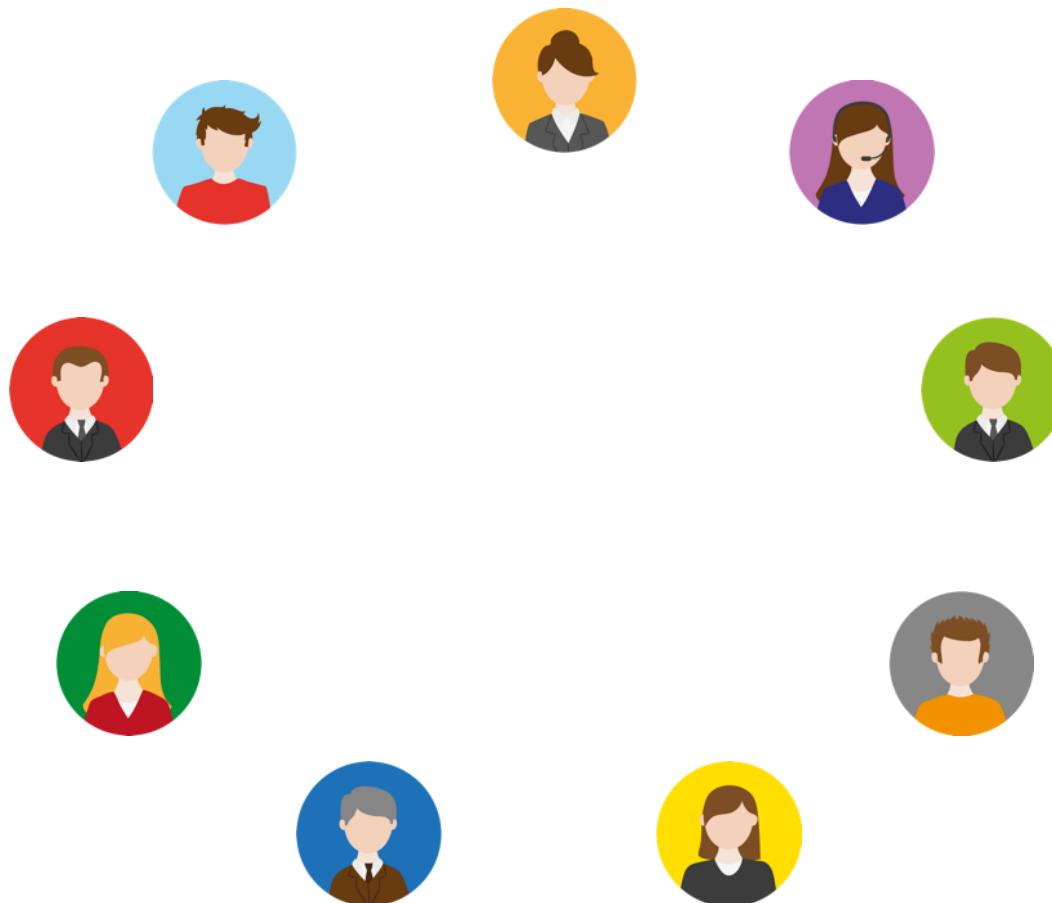
# Our First Protocol: Private Sum



# Security

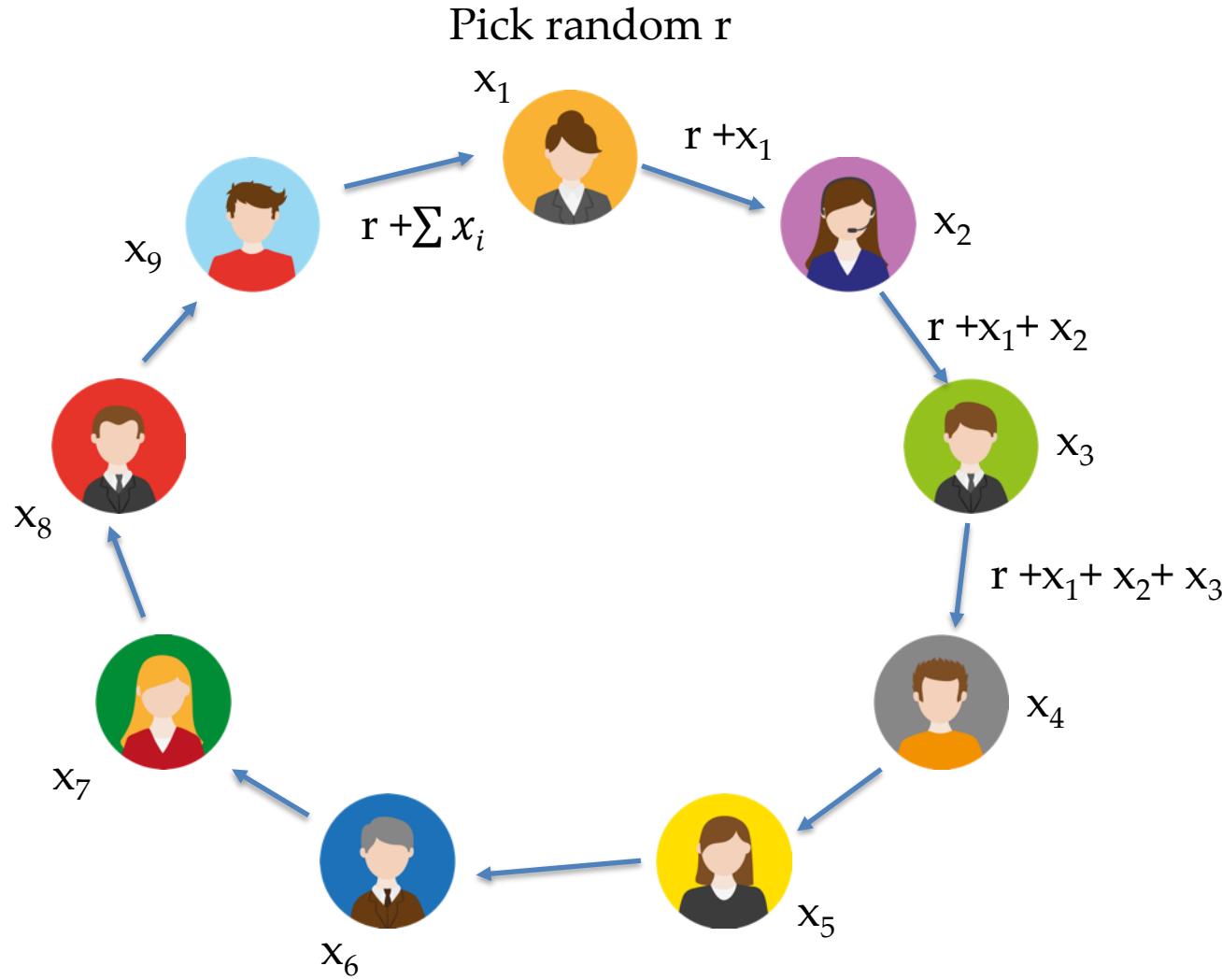


# Example Protocol



# (In)Security

The protocol is  
not secure if ....



# Classifications

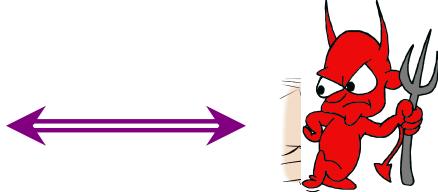
- How many parties can collude?
  - Honest majority, dishonest majority
- What can corrupted parties do?
  - Semi-honest; malicious
- How many parties are guaranteed to obtain output?
  - Security with abort, fairness, Guaranteed output delivery

# Definition

## Implication

Anything that an adversary could have learned/done in the real model, it could have also learned/done in the ideal model.

For every real adversary **A**



Protocol interaction

there exists an adversary **S**



Trusted party

**REAL**

**IDEAL**

# Common Building blocks

- Basic tools:
  - Garbled Circuit, Oblivious Transfer, Beaver Triple, Secret Sharing
- More complicated tools:
  - Private set intersection, function secret-sharing, RAM-based secure computation

# More Building blocks

- Private information retrieval
- Fully homomorphic encryption
- Zero-knowledge proof

# MPC Materials

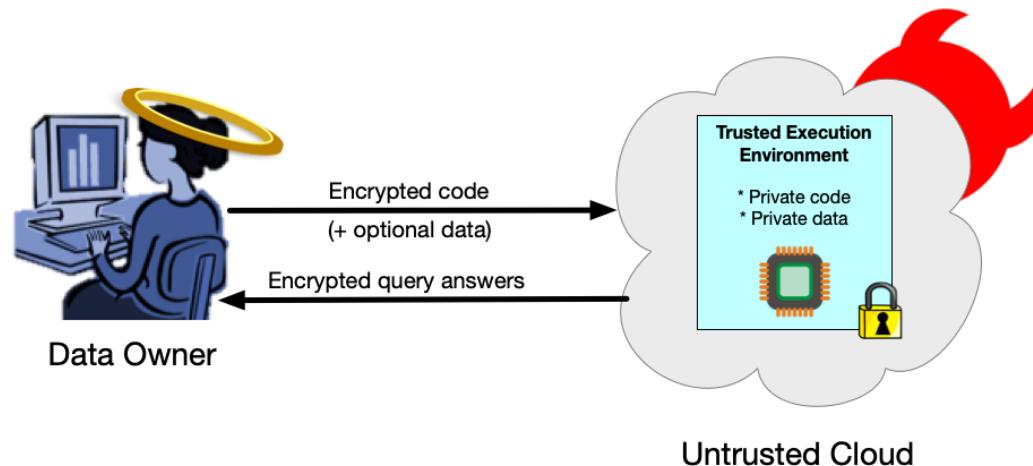
You can find almost all at <https://github.com/rdragos/awesome-mpc>

- Video Lectures
  - [1<sup>st</sup>,5th] BIU Winter School
- Open-source libs: first think about what you are looking for
  - How many parties?
  - What security model?
  - What programming language?

# TRUSTED EXECUTION ENVIRONMENTS

# Trusted Execution Environment

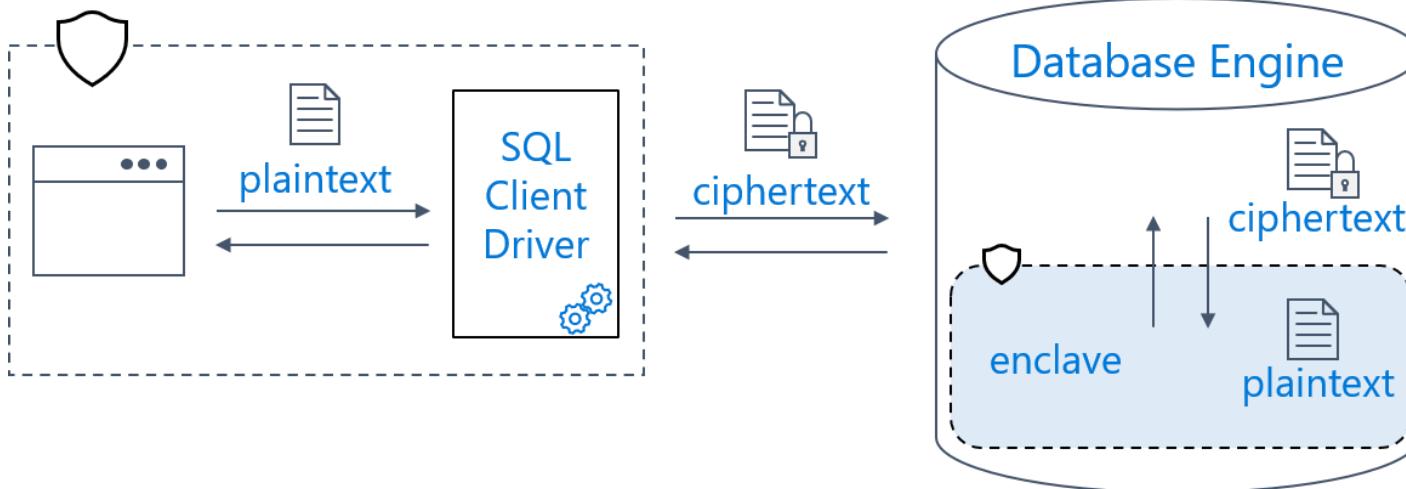
- Confidential computing for untrusted, remote hosts



Offers integrity guarantees too.

# TEE Use Cases

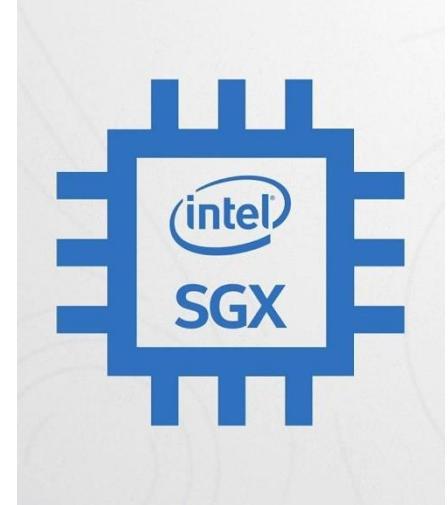
-  Authentication on mobile devices
-  Digital rights management
-  Outsourcing business ops to the cloud



Microsoft SQL Server Always Encrypted Workflow

# TEEs are everywhere...

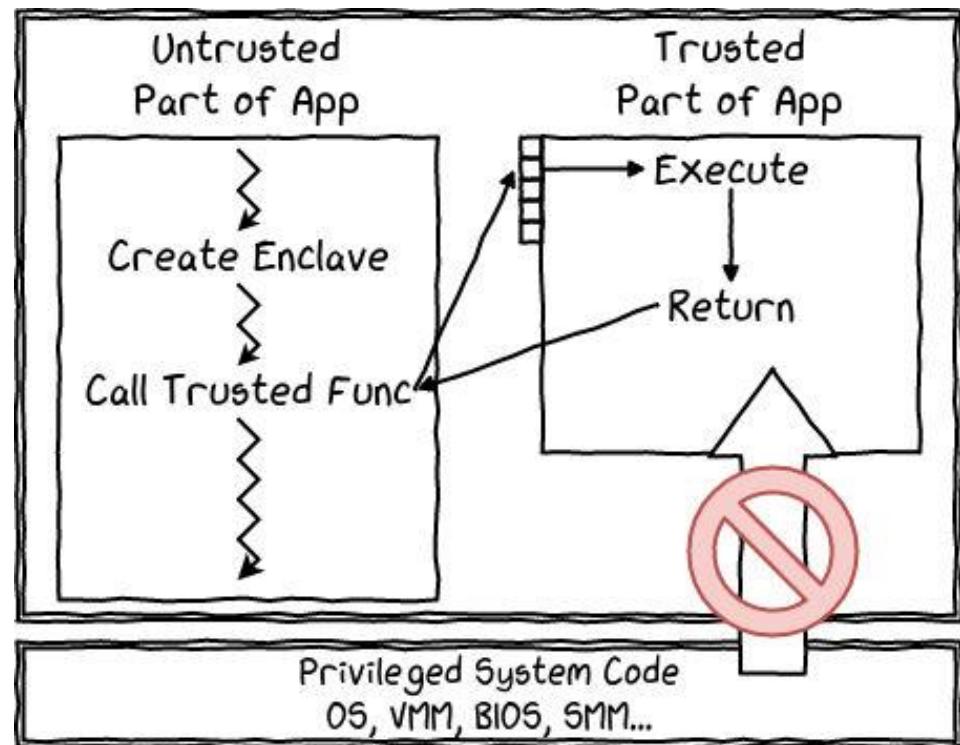
- Intel SGX
- ARM TrustZone
- AWS Nitro Enclaves
- Apple Secure Enclave Processor
- Keystone Enclave



Related: AMD Secure Encrypted Virtualization (used by GCP)

# TEE Application Setup

- DBMS partitioned into trusted and untrusted code
- Private data and app logic are sealed into enclave
- Enclave may read from untrusted app, but not vice versa!



# TEE Features

Confidentiality for process and data.

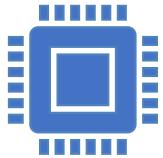
Secure communication channels to remote hosts

Integrity: run on trusted hardware alone

# TEE Building Blocks

- Attestation – verify enclave on untrusted host
- Encrypted Page Cache (EPC) – memory accessible to enclave alone
- Instruction set (ISA) extensions





# DBMS Design Decisions on TEEs

- Defining security guarantees for system
- Determines what parts of app need to be in enclave
- Partition data so that it fits in protected memory

# TEE Design Pitfalls

- Expensive to move data in and out of EPC
  - EPC is small  $\sim O(100 \text{ MB})$
- No systematic way to partition a program's native and enclave code for crypto-strong guarantees
- Privacy-performance trade-offs

# Example design questions for DBMS-over-TEE

- Should we protect the cardinalities of intermediate results?
- How do we make swapping data in and out of the EPC data-independent?
- How to not leak information with index lookups and writes?
- Should we make our queries private or our computation thereof alone?

# TEE Vulnerabilities

- TEEs – like all hardware – are susceptible to cycles of finding new attacks and mitigating them.
- Sometimes need to fix attacks in silica.
- They remain valuable platforms for research.



**CacheOut**  
Leaking Data on Intel CPUs via Cache Evictions

We present CacheOut, a new speculative execution attack that is capable of leaking data from Intel CPUs across many security boundaries. We show that despite Intel's attempts to address previous generations of speculative execution attacks, CPUs are still vulnerable, allowing attackers to exploit these vulnerabilities to leak sensitive data.

Moreover, unlike previous MDS issues, we show in our work how an attacker can exploit the CPU's caching mechanisms to select what data to leak, as opposed to waiting for the data to be available. Finally, we empirically demonstrate that CacheOut can violate nearly every hardware-based security domain, leaking data from the OS kernel, co-resident virtual machines, and even SGX enclaves.

[Read the Paper](#) [Cite](#)

**SGAxe**  
How SGX Fails in Practice

SGAxe is an evolution of CacheOut, specifically targeting SGX enclaves. We show that despite extensive efforts done by Intel in order to mitigate SGX side channels, an attacker can still breach the confidentiality of SGX enclaves even when all side channel countermeasures are enabled.

We then proceed to show an extraction of SGX private attestation keys from within SGX's quoting enclave, as compiled and signed by Intel. With these keys in hand, we are able to sign fake attestation quotes, just as if these have initiated from trusted and genuine SGX enclaves. This erodes trust in the SGX ecosystem, as using such quotes an attacker can masquerade itself as a genuine SGX enclave to a remote party, while offering little protection in reality.

[Read the Paper](#) [Cite](#)

# Pros and Cons of TEEs

Pros:

- Efficient performance
- May parallelize over a set of enclaves

Cons:

- Vendor lock-in
- Side-channel leakage
- Security guarantees brittle since hardware fixes are slow

# MODULE 2B

# DIFFERENTIAL PRIVACY

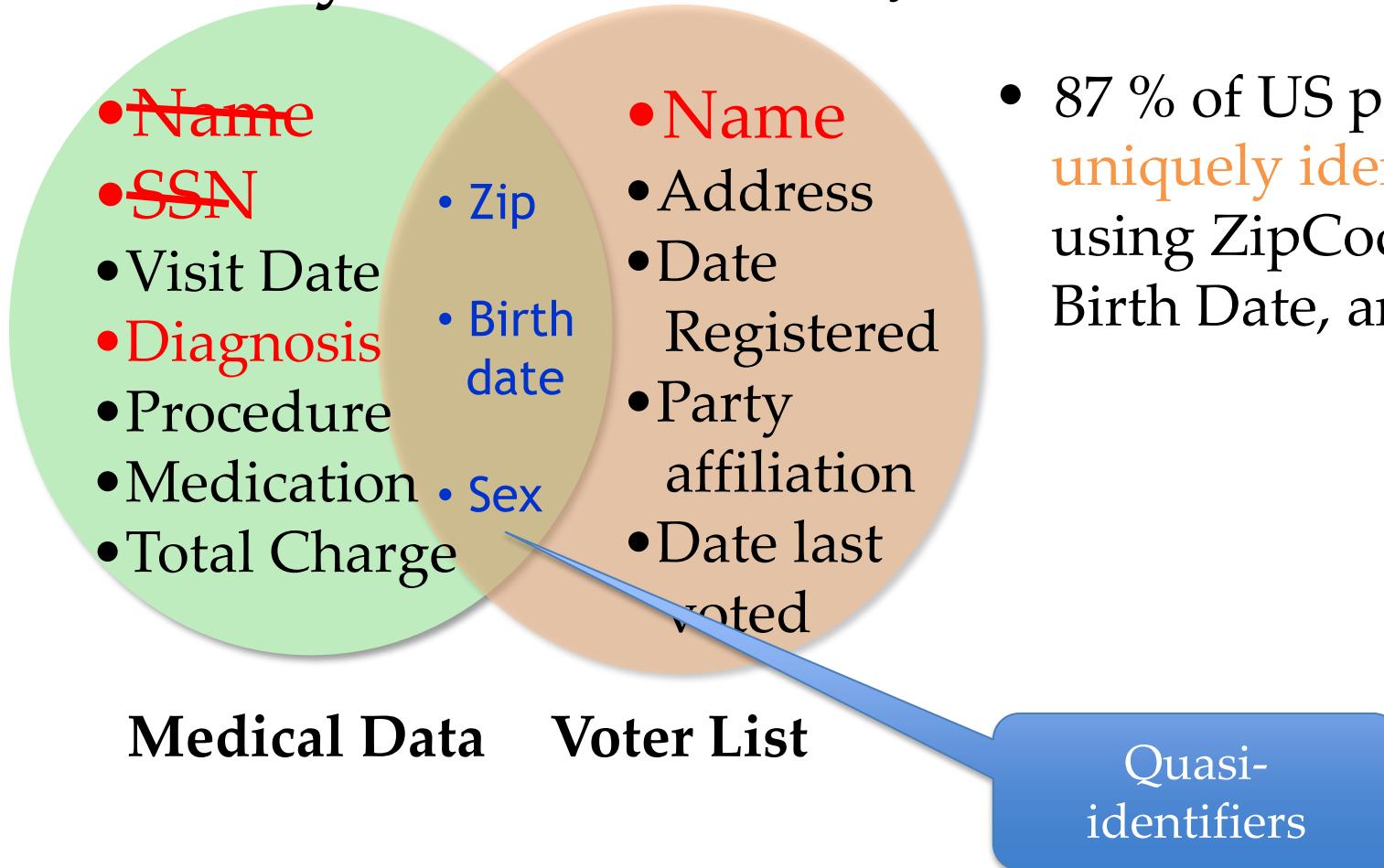


# Outline

- Desiderata for Defining Privacy
- Differential Privacy (DP) Basics
- Integration of DP into DB & Challenges



# The Massachusetts Governor Privacy Breach [Sweeney IJUFKS 2002]



# K-Anonymity: Avoiding Linkage Attacks

[S 02]

- If every row corresponds to one individual ...  
... every row should look like  $k-1$  other rows based on the *quasi-identifier* attributes



# K-Anonymity

Zip	Age	Nationality	Disease
13053	28	Russian	Heart
13068	29	American	Heart
13068	21	Japanese	Flu
13053	23	American	Flu
14853	50	Indian	Cancer
14853	55	Russian	Heart
14850	47	American	Flu
14850	59	American	Flu
13053	31	American	Cancer
13053	37	Indian	Cancer
13068	36	Japanese	Cancer
13068	32	American	Cancer



Zip	Age	Nationality	Disease
130**	<30	*	Heart
130**	<30	*	Heart
130**	<30	*	Flu
130**	<30	*	Flu
1485*	>40	*	Cancer
1485*	>40	*	Heart
1485*	>40	*	Flu
1485*	>40	*	Flu
130**	30-40	*	Cancer
130**	30-40	*	Cancer
130**	30-40	*	Cancer
130**	30-40	*	Cancer



# Problem 1: Background knowledge

Adversary knows prior knowledge about Umeko

Adversary learns Umeko has Cancer

Name	Zip	Age	Nat.
Umeko	13053	25	Japan

Zip	Age	Nationality	Disease
130**	<30	*	Heart
130**	<30	*	Heart
130**	<30	*	Cancer
130**	<30	*	Cancer
1485*	>40	*	Cancer
1485*	>40	*	Heart
1485*	>40	*	Flu
1485*	>40	*	Flu
130**	30-40	*	Cancer
130**	30-40	*	Cancer
130**	30-40	*	Cancer
130**	30-40	*	Cancer



# Attacks using Background Knowledge

- Record-level Data
  - Netflix Data [[Narayanan-Shmatikov, 2008]]
- Search Logs
  - AOL data publishing [Barbaro-Zeller, 2006]
- Graph/Social Network Data
  - Degrees of nodes [Liu and Terzi, SIGMOD 2008]
  - The network structure, e.g., a subgraph of the network.  
[Zhou and Pei, ICDE 2008, Hay et al., VLDB 2008]



# Desiderata for a Privacy Definition

## 1. Resilience to background knowledge

- A privacy mechanism must be able to protect individuals' privacy from attackers who may possess background knowledge



# Problem 2: Privacy by Obscurity

- Many organization think their data are private because they perturb the data and make the parameters of perturbation secret.



# Desiderata for a Privacy Definition

## 1. Resilience to background knowledge

- A privacy mechanism must be able to protect individuals' privacy from attackers who may possess background knowledge

## 2. Privacy without obscurity

- Attacker must be assumed to know the algorithm used as well as all parameters [MK15]



# Problem 3: Post-processing

 U.S. Department of Health & Human Services

[About Us](#) [Careers](#) [Contact Us](#) [Español](#) [FAQ](#) [!\[\]\(d67558004ec91de7e7737b0c029eefb9\_img.jpg\) Email Updates](#)



[Home](#) [Glossary](#) [Methodology](#) [Our Partners](#) [Tutorial](#)

## Free Health Care Statistics

HCUPnet is a free, on-line query system based on data from the Healthcare Cost and Utilization Project (HCUP)

The system provides health care statistics and information for hospital inpatient, emergency department, and ambulatory settings, as well as population-based health care data on counties

[Create a New Analysis](#) 

[Get Quick Statistics Tables](#) 

[Find out more about HCUP](#)

[What's new with HCUPnet](#)

The HCUPnet Web site has been redesigned. The new site has a modernized look and feel, a simplified process for querying data, fewer clicks to reach the same information, and more flexibility in changing the content and display of data you are viewing.



# Problem 3: Post-processing

Counts less than k are suppressed  
achieving k-anonymity

Age	#discharges	White	Black	Hispanic	Asian/Pcf Hlnder	Native American	Other	Missing
#discharges	735	535	82	58	18	*	19	22
1-17	*	*	*	*	*	*	*	*
18-44	70	40	13	*	*	*	*	*
45-64	330	236	31	32	*	*	11	*
65-84	298	229	35	13	*	*	*	*
85+	34	29	*	*	*	*	*	*



# Problem 3: Post-processing

Age	#discharges	White	Black	Hispanic	Asian/Pcf Hlnder	Native American	Other	Missing
#discharges	735	535	82	58	18	1	19	22
1-17	3	1	*	*	*	*	*	*
18-44	70	40	13	*				
45-64	330	236	31	32				
65-84	298	229	35	13	*	*	*	*
85+	34	29	*	*	*	*	*	*

$$= 535 - (40+236+229+29)$$



# Desiderata for a Privacy Definition

## 1. Resilience to background knowledge

- A privacy mechanism must be able to protect individuals' privacy from attackers who may possess background knowledge

## 2. Privacy without obscurity

- Attacker must be assumed to know the algorithm used as well as all parameters [MK15]

## 3. Post-processing

- Post-processing the output of a privacy mechanism must not change the privacy guarantee [KL10, MK15]



# Problem 4: Multiple Releases

- 2 tables of k-anonymous patient records

	Non-Sensitive			Sensitive Condition
	Zip code	Age	Nationality	
1	130**	<30	*	AIDS
2	130**	<30	*	Heart Disease
3	130**	<30	*	Viral Infection
4	130**	<30	*	Viral Infection
5	130**	≥40	*	Cancer
6	130**	≥40	*	Heart Disease
7	130**	≥40	*	Viral Infection
8	130**	≥40	*	Viral Infection
9	130**	3*	*	Cancer
10	130**	3*	*	Cancer
11	130**	3*	*	Cancer
12	130**	3*	*	Cancer

Hospital A (4-anonymous)

	Non-Sensitive			Sensitive Condition
	Zip code	Age	Nationality	
1	130**	<35	*	AIDS
2	130**	<35	*	Tuberculosis
3	130**	<35	*	Flu
4	130**	<35	*	Tuberculosis
5	130**	<35	*	Cancer
6	130**	<35	*	Cancer
7	130**	≥35	*	Cancer
8	130**	≥35	*	Cancer
9	130**	≥35	*	Cancer
10	130**	≥35	*	Tuberculosis
11	130**	≥35	*	Viral Infection
12	130**	≥35	*	Viral Infection

Hospital B (6-anonymous)

- If Alice visited both hospitals and she is 28, can you deduce Alice's medical condition?



# Problem 4: Multiple Releases

- 2 tables of k-anonymous patient records [GKS08]

	Non-Sensitive			Sensitive Condition
	Zip code	Age	Nationality	
1	130**	<30	*	AIDS
2	130**	<30	*	Heart Disease
3	130**	<30	*	Viral Infection
4	130**	<30	*	Viral Infection
5	130**	≥40	*	Cancer
6	130**	≥40	*	Heart Disease
7	130**	≥40	*	Viral Infection
8	130**	≥40	*	Viral Infection
9	130**	3*	*	Cancer
10	130**	3*	*	Cancer
11	130**	3*	*	Cancer
12	130**	3*	*	Cancer

Hospital A (4-anonymous)

- 2 tables of k-anonymous patient records [GKS08]

- Alice is 28 and she visits both hospitals
- 4-anonymity + 6-anonymity  $\not\Rightarrow$  k-anonymity , for any k

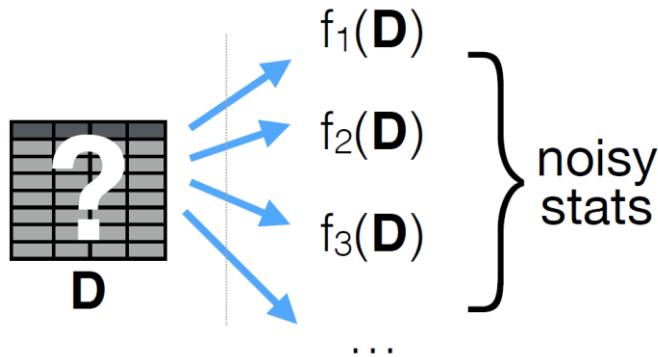
Hospital B (6-anonymous)

- Alice is 28 and she visits both hospitals
- 4-anonymity + 6-anonymity  $\not\Rightarrow$  k-anonymity , for any k



# Database Reconstruction Theorem

- Informally: If *too many statistics* are released *too accurately*, the vast majority of the records in the (hidden) database can be *reconstructed*.



## Reconstruction attack

$D$  is unknown, find  $D$  that best matches released statistics

Successfully demonstrated by US Census Bureau in 2019.



# A Bound on the Number of Queries

- In order to ensure utility, a statistical database must leak some information about each individual
- We can only hope to bound the amount of disclosure
- Hence, there is a limit on number of queries that can be answered



# Desiderata for a Privacy Definition

## 1. Resilience to background knowledge

- A privacy mechanism must be able to protect individuals' privacy from attackers who may possess background knowledge

## 2. Privacy without obscurity

- Attacker must be assumed to know the algorithm used as well as all parameters [MK15]

## 3. Post-processing

- Post-processing the output of a privacy mechanism must not change the privacy guarantee [KL10, MK15]

## 4. Composition over multiple releases

- Allow a graceful degradation of privacy with multiple invocations on the same data [DN03, GKS08]

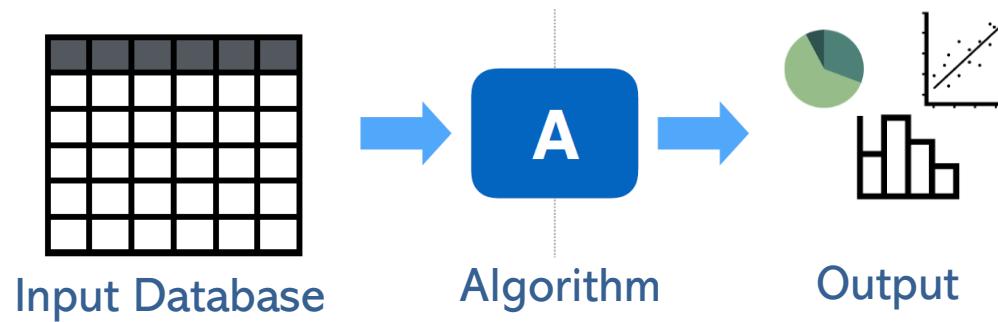


# Outline

- Desiderata for Defining Privacy
- Differential Privacy (DP) Basics
- Integration of DP into DB & Challenges



# Differential Privacy



- Differential privacy is *not* an algorithm
- Differential privacy is *not* a property of the output
- Differential privacy *is* a property of the algorithm



# Illustrative Application

name	gen.	age	HR	BP	...
Alice	F	83	65	112	...
Bob	M	50	85	135	...
Carl	M	23	61	120	...
...	...	...	...	...	...

Data from a medical study  
attributes include age, heart rate (HR),  
blood pressure (BP), results of various  
medical tests

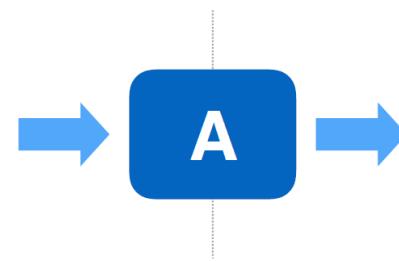
Analyst  
identify risk of heart disease  
for different demographic  
groups



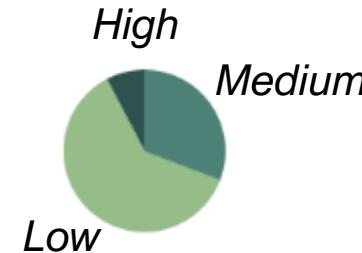
# Illustrative Application

name	gen.	age	HR	BP	...
Alice	F	83	65	112	...
Bob	M	50	85	135	...
Carl	M	23	61	120	...
...	...	...	...	...	...

Input D



Algorithm A



Output A(D)



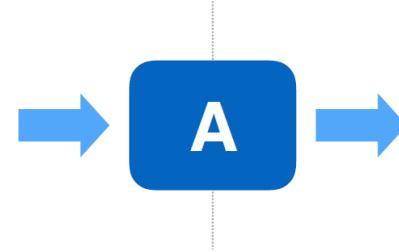
*Estimated risk of  
heart disease for  
males, 40-50 yrs old  
HR in 60-85*



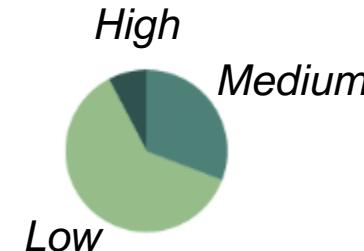
# Privacy Risk of Releasing $A(D)$

name	gen.	age	HR	BP	...
Alice	F	83	65	112	...
Bob	M	50	85	135	...
Carl	M	23	61	120	...
...	...	...	...	...	...

Input D



Algorithm A



Output  $A(D)$



Bob

*Could  $A(D)$  reveal my risk of heart disease?  
Could this lead to an increase in my insurance premium?*



# Defining Privacy: Attempt 1

- Mechanism is private if an attacker can not learn too much about an individual beyond what they already know about the individual (without looking at the output of the mechanism)

*NOT A GOOD DEFINITION*

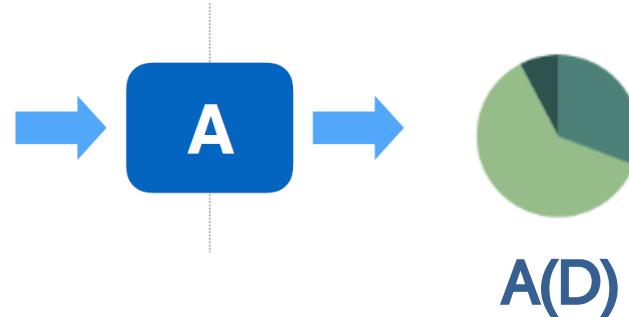
- Learning principles of nature should not be considered privacy breaches.



# Defining Privacy: Attempt 2

nam	gen.	age	HR	...
Alice	F	83	65	...
Bob	M	50	85	...
Carl	M	23	61	...
...	...	...	...	...

D



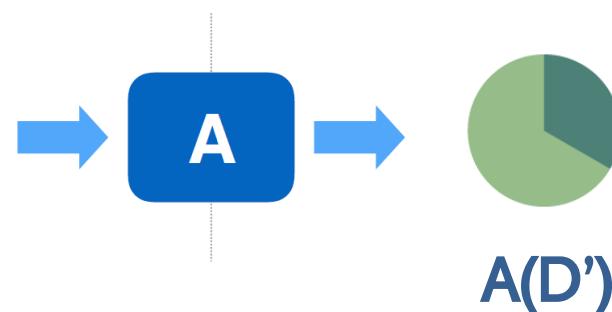
Real world

Promise of differential privacy

*“What can be learned about Bob from  $A(D)$  is similar to what can be learned from opt-out world”*

nam	gen.	age	HR	...
Alice	F	83	65	...
XXX	XXX	XXX	XXX	...
Carl	M	23	61	...
...	...	...	...	...

D'



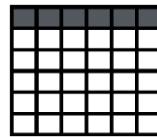
Bob's opt-out world



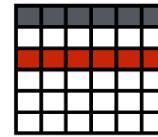
# Differential Privacy

[Dwork ICALP 2006]

For every pair of inputs that differ in one row



$D_1$



$D_2$

For every output ...



$O$

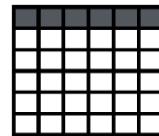
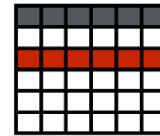
Adversary should not be able to distinguish between any  $D_1$  and  $D_2$  based on any  $O$

$$\ln \left( \frac{\Pr[A(D_1) = o]}{\Pr[A(D_2) = o]} \right) \leq \varepsilon, \quad \varepsilon > 0$$



# Why pairs of datasets that *differ in one row*?

For every pair of inputs that differ in one row

 $D_1$  $D_2$ 

For every output ...

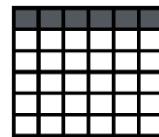
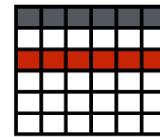
 $O$ 

Simulate the presence or absence of a single record



# Why *all* pairs of datasets ...?

For every pair of inputs that differ in one row

 $D_1$  $D_2$ 

For every output ...

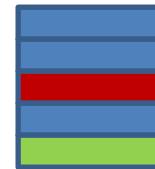
 $O$ 

Guarantee holds no matter what the other records are.

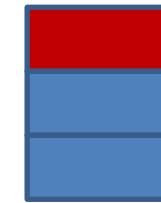


# Privacy Parameter $\varepsilon$

For every pair of inputs that differ in one row


 $D_1$ 

 $D_2$ 

For every output ...


 $O$ 

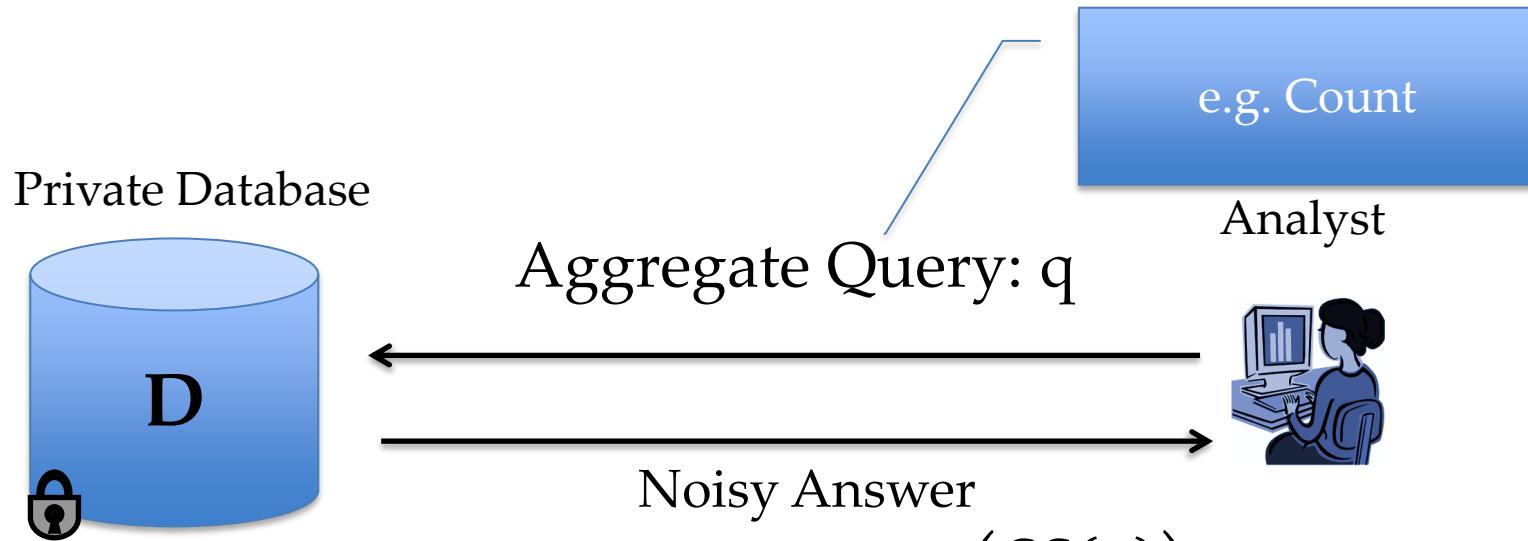
$$\ln \left( \frac{\Pr[A(D_1) = o]}{\Pr[A(D_2) = o]} \right) \leq \varepsilon, \quad \varepsilon > 0$$

Controls the degree to which  $D_1$  and  $D_2$  can be distinguished.  
Smaller the  $\varepsilon$  more the privacy (and worse the utility)

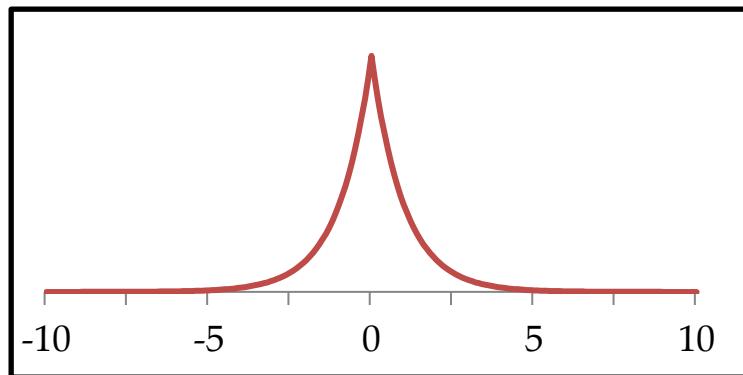


# Laplace Mechanism

[DMNS 06]



$$\tilde{q}(D) = q(D) + \text{Lap}\left(\frac{GS(q)}{\varepsilon}\right)$$

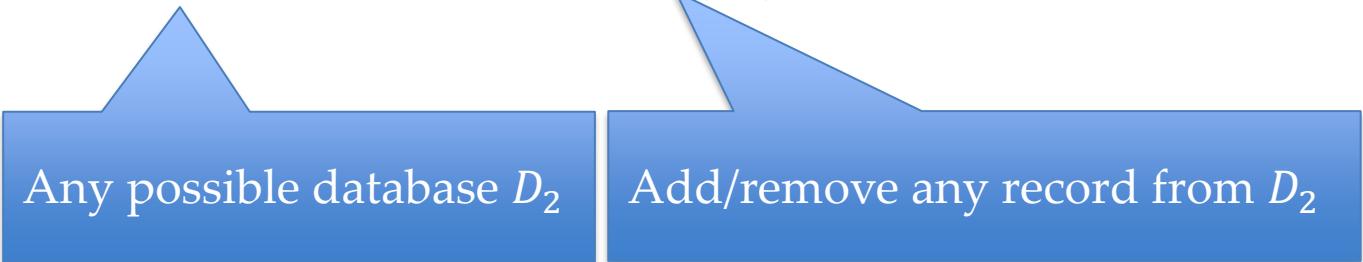


$$\text{Lap}(\lambda): h(\eta) \propto \exp\left(-\frac{|\eta|}{\lambda}\right)$$

# How much noise for privacy?

**Global Sensitivity** of a query  $q$ : maximum output change for any pairs of neighboring datasets

$$\begin{aligned} GS(q) &= \max_{\forall \text{neighbor}(D_1, D_2)} |q(D_1) - q(D_2)| \\ &= \max_{D_2 \in \text{dom}} \max_{\forall D_1 \in \text{neighbors}(D_2)} |q(D_1) - q(D_2)| \end{aligned}$$



Any possible database  $D_2$

Add/remove any record from  $D_2$

Theorem:  $q(D) + \text{Lap}\left(\frac{GS(q)}{\varepsilon}\right)$  satisfies  $\varepsilon$ -DP.



# Global Sensitivity: COUNT query

- # of people having flu?
- Global sensitivity = 1

Sex	Height	Age	Disease	Drug X
M	6'2"	56	Cancer	3.5
F	5'3"	30	Diabetes	2.3
F	5'9"	24	Healthy	1.0
M	5'3"	36	Flu	4.0
M	6'7"	22	Flu	2.2

- Solution:  $2 + \eta$ , where  $\eta$  is drawn from  $Lap(\frac{1}{\epsilon})$ 
  - Mean = 0
  - Variance =  $2/\epsilon^2$



# Global Sensitivity: SUM query

- Total usage of drug X?

Sex	Height	Age	Disease	Drug X
M	6'2"	56	Cancer	3.5
F	5'3"	30	Diabetes	2.3
F	5'9"	24	Healthy	1.0
M	5'3"	36	Flu	4.0
M	6'7"	22	Flu	2.2

- Suppose all values  $x$  are in  $[a,b]$
- Global sensitivity =  $b - a$

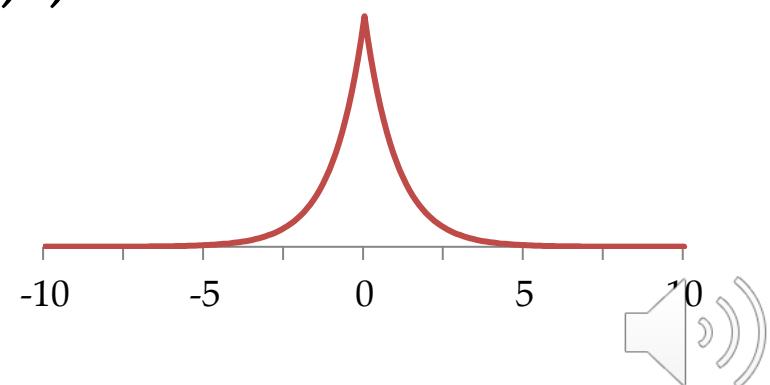


# Utility of Laplace Mechanism

- Laplace mechanism works for any function that returns a real number
- Error:  $E(\text{true answer} - \text{noisy answer})^2$

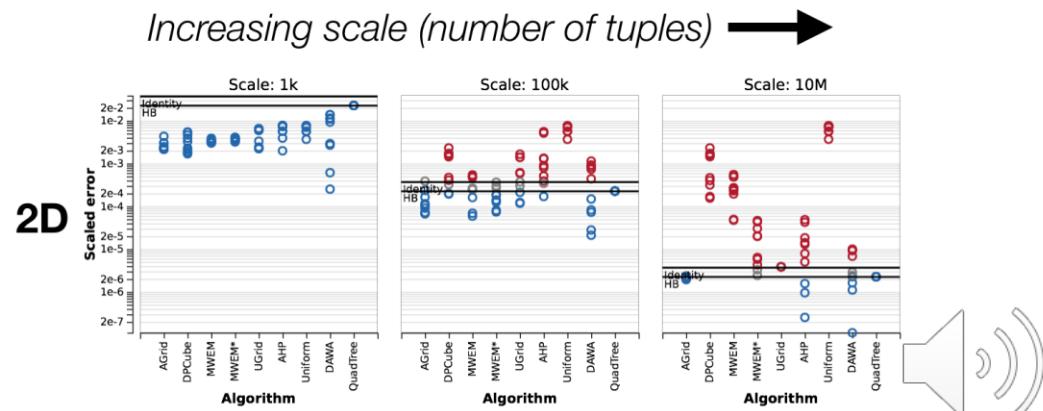
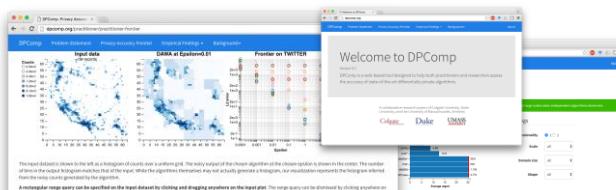
$$= \text{Var}(\text{Lap}(\text{GS}(q)/\varepsilon))$$

$$= 2 * \text{GS}(q)^2 / \varepsilon^2$$



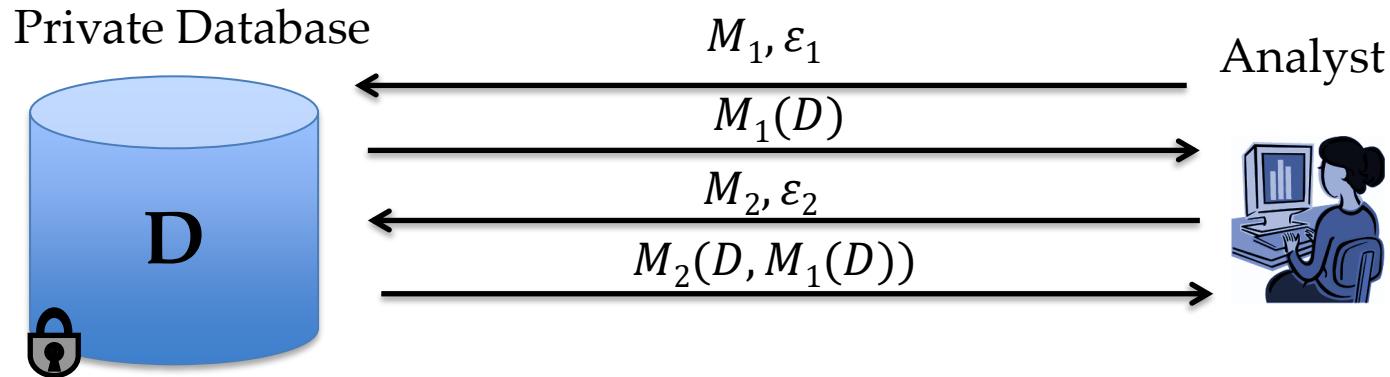
# Accuracy-privacy Trade-offs

- Many DP algorithms:
  - Laplace mechanism, exponential mechanism, randomized response, gaussian mechanism, sample and aggregate, report noisy max, sparse vector technique, smooth sensitivity mechanism,.....
- Each gives a different accuracy-privacy trade-off
  - e.g. DPComp [HMMCZ16]



# Sequential Composition

[DMNS 06]



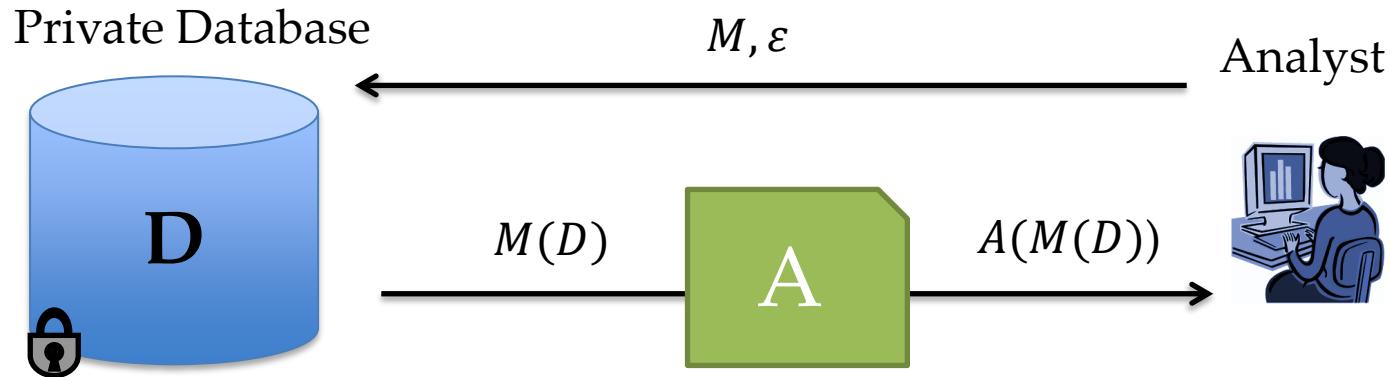
If  $M_1, M_2, \dots, M_k$  are algorithms that access a private database D such that each  $M_i$  satisfies  $\varepsilon_i$ -DP

then the combination of their outputs satisfies  $\varepsilon$ -DP with  $\varepsilon = \varepsilon_1 + \dots + \varepsilon_k$



# Postprocessing

[DMNS 06]



If  $M$  is an  $\varepsilon$ -differentially private algorithm, then any additional post-processing  $A \circ M$  also satisfies  $\varepsilon$ -differential privacy.



# DP in Practice

The collage includes the following elements:

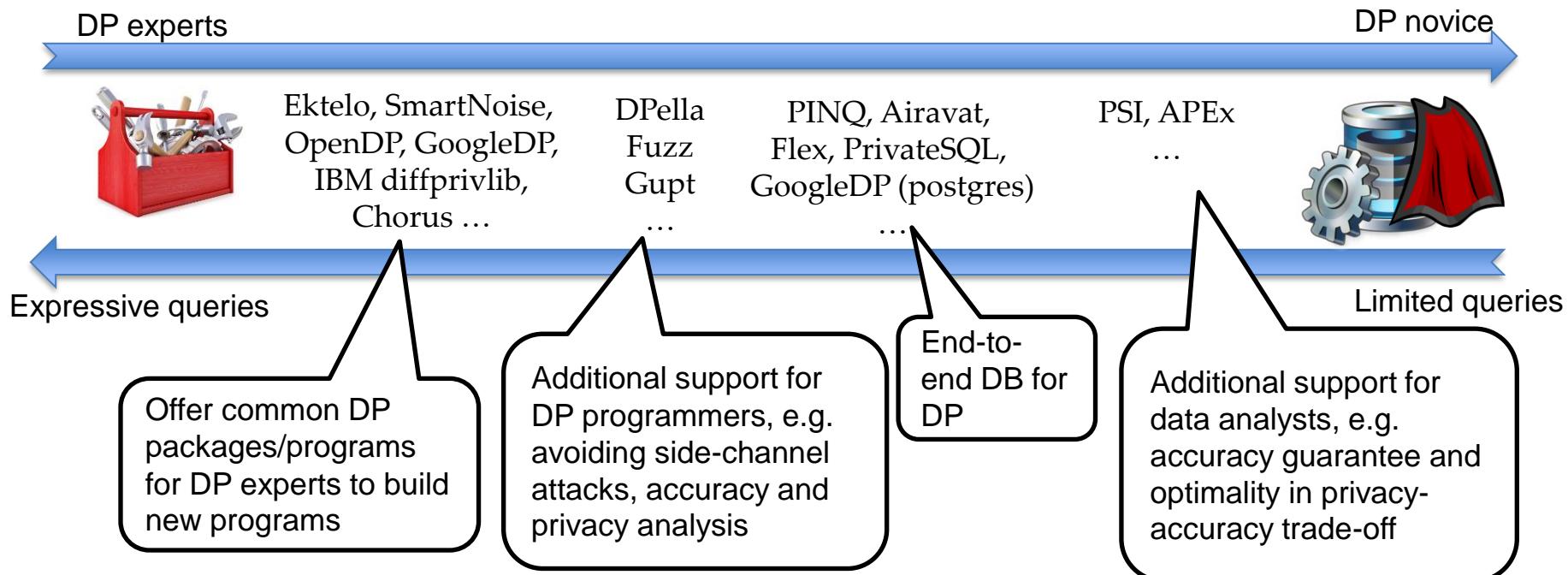
- OnTheMap**: A screenshot of a geographic analysis tool showing a map of the United States with state boundaries. A green callout box contains the text: "Synthetic data about where people in the US live and work".
- United States Census Bureau**: A logo featuring the text "United States Census Bureau" over a map of the United States.
- Learning Popular Emojis with Privacy**: A histogram titled "Frequency" showing the distribution of emoji usage.
- ECG/EKG**: A green ECG/EKG signal plotted against a grid.
- Windows Taskbar**: A screenshot of a Windows taskbar showing several pinned icons.
- Google Chrome Settings**: A screenshot of the Google Chrome settings page, specifically the "Privacy and security" section, showing checkboxes for "Automatically send usage statistics", "Send RAPPORT statistics to Google", and "Send a 'Do Not Track' request".
- Google Logo**: The classic multi-colored Google logo.
- How Google is Using Differential Privacy for COVID Location Data**: A title card with a blue background. Below it is a paragraph of text explaining Google's use of Differential Privacy during COVID-19 lockdowns, followed by three line graphs showing mobility trends for different locations (Retail & Recreation, Grocery & Pharmacy, Parks) comparing March 2020 to March 2021. The graphs show significant decreases in activity levels.
- Click on this box to learn more.**: A purple button with white text.

<http://onthemap.ces.census.gov/>

<https://www.recurve.com/blog/traditional-approaches-to-protecting-energy-data-dont-work-heres-what-to-do-instead-part-3-of-3>



# Tools & Systems for DP



# Outline

- Desiderata for Defining Privacy
- Differential Privacy (DP) Basics
- Integration of DP into DB & Challenges



# Engineering DP into DBMS

- Existing DP database systems:
  - PINQ[SIGMOD09], Airavat[NSDI10], Flex(Uber DP)[VLDB18], Google DP[19]
- Rule-based sensitivity analysis of a query plan



# What could go wrong?

- Standard SQL queries that cannot be answered accurately
  - E.g., Non-aggregate/Max/Min query
- Group-by in the end of the query

Person					
ID	Sex	Age	...	HID	Geo
122	M	40	...	H6	CA
123	F	12	...	H6	CA
124	M	23	...	H7	FL
125	M	26	...	H8	NC
126	F	30	...	H8	NC

Select Geo, Count() from Person  
Group By Geo;

Geo	Count
CA	2
FL	1
NC	2

- Leak active domain!



# DB Queries for DP

- Existing solutions:
  - Specify groupby keys or Use partition (all groups)
- Open questions:
  - Private data-manipulation language (PDML)
    - How to handle correlated subqueries?

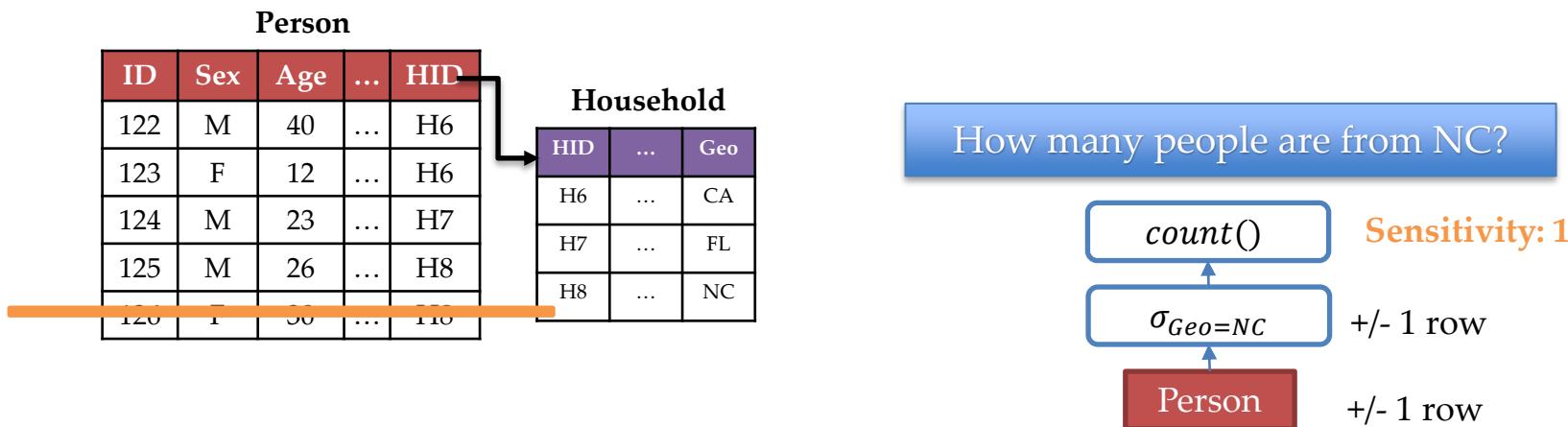
```
SELECT relp, race, cnt FROM Person P, (SELECT COUNT(*) AS cnt, hid FROM  
PERSON GROUP BY hid) AS P2 WHERE P2.hid=P.hid
```

- Additional specification of budget or accuracy



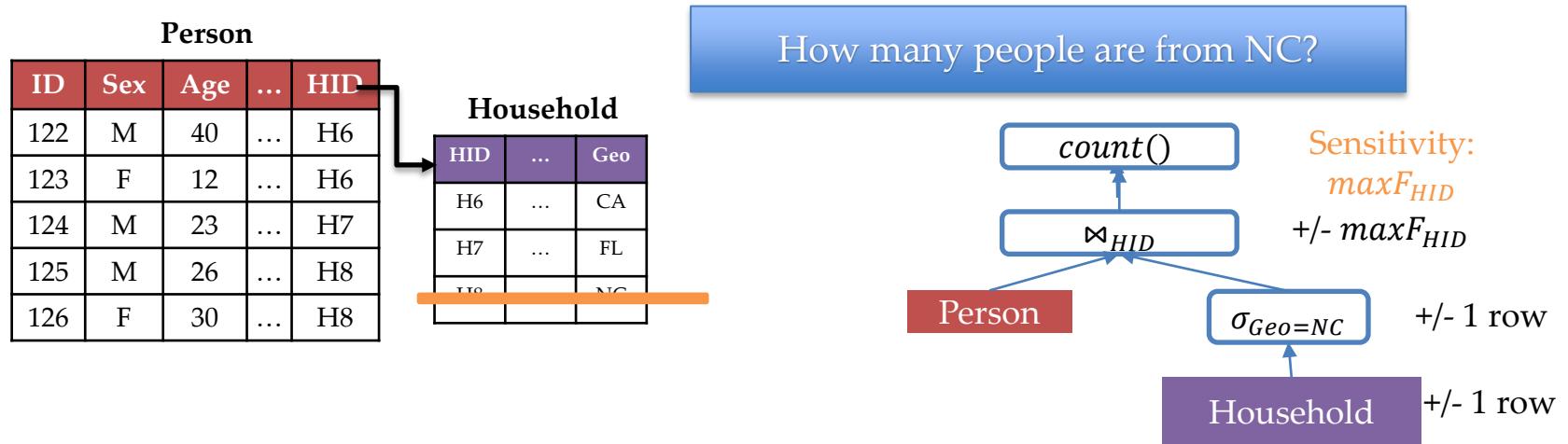
# What could go wrong?

- If storing multiple tables instead?



# What could go wrong?

- If storing multiple tables instead?
  - Adding/removing rows of different tables  
→ different sensitivity result

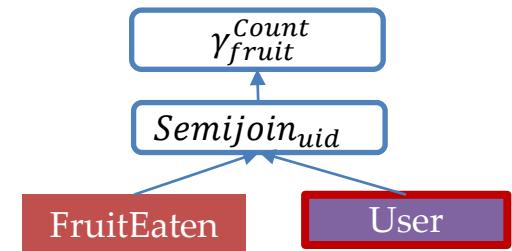


# Defining Private Objects

PINQ	GoogleDP	Flex	PrivateSQL
One-row (single policy)	User-level (single policy)	One-row (single policy)	Constraints-based (multiple policies)

```
SELECT fruit, COUNT(fruit)
FROM FruitEaten
GROUP BY fruit;
```

```
SELECT result.fruit, result.number_eaten
FROM (
    SELECT per_person.fruit,
        ANON_SUM(per_person.fruit_count, LN(3)/2) as number_eaten,
        ANON_COUNT(uid, LN(3)/2) as number_eaters
    FROM(
        SELECT * , ROW_NUMBER() OVER (
            PARTITION BY uid
            ORDER BY random()
        ) as row_num
        FROM (
            SELECT fruit, uid, COUNT(fruit) as fruit_count
            FROM FruitEaten
            GROUP BY fruit, uid
        ) as per_person_raw
        ) as per_person
    WHERE per_person.row_num <= 5
    GROUP BY per_person.fruit
) as result
WHERE result.number_eaters > 50;
```



<https://github.com/google/differential-privacy/tree/main/cc/postgres>



# Defining Private Objects



**Edge-privacy:** hide the presence of an edge (a row in edge table)

**Node-privacy:** hide the presence of a node and all edges incident to it. (a row in node table + edges)

**Person-privacy:** hide properties of people  
**Household-privacy:** hide properties of households and the people within them.

Person			
ID	Sex	...	HID
122	M	...	H6
123	F	...	H6
124	M	...	H7
125	M	...	H8
126	F	...	H8

Household		
HID	...	Geo
H6	...	CA
H7	...	FL
H8	...	NC

**Policy:** A specification of the base relation that is **the primary private object**

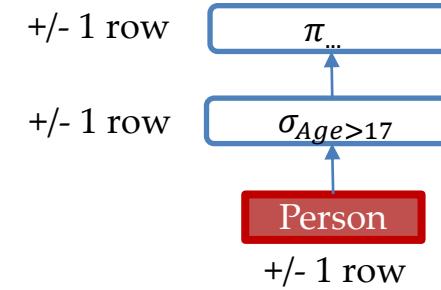
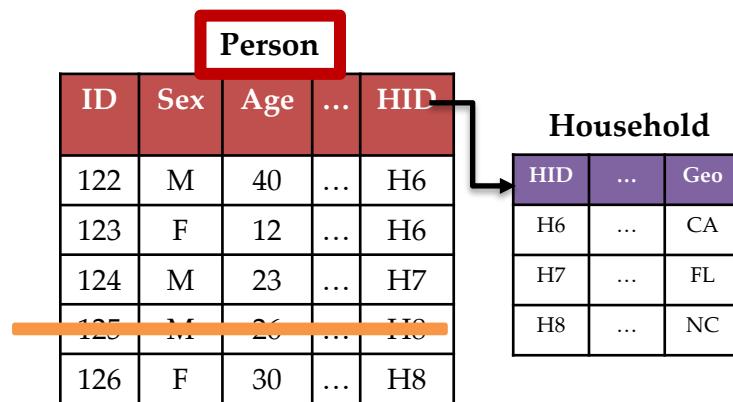
- Key technical insight: leverage foreign key constraints to infer how change to primary table affects sensitivity of query (even a query that does not directly involve primary table!)
- Current limitation: foreign key constraints must be *acyclic*



# Defining Private Objects

- Policy: Person

V:= SELECT \* FROM PERSON WHERE PERSON.Age>17;



# Defining Private Objects

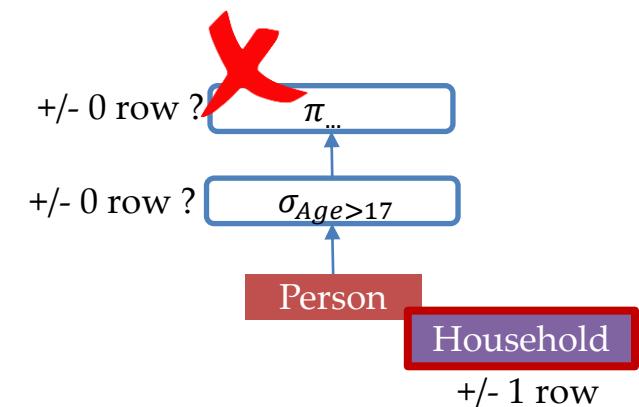
- Policy: Household

V:= SELECT \* FROM PERSON WHERE PERSON.Age>17;

Person				
ID	Sex	Age	...	HID
122	M	40	...	H6
123	F	12	...	H6
124	M	23	...	H7
125	M	26	...	H8
126	F	30	...	H8

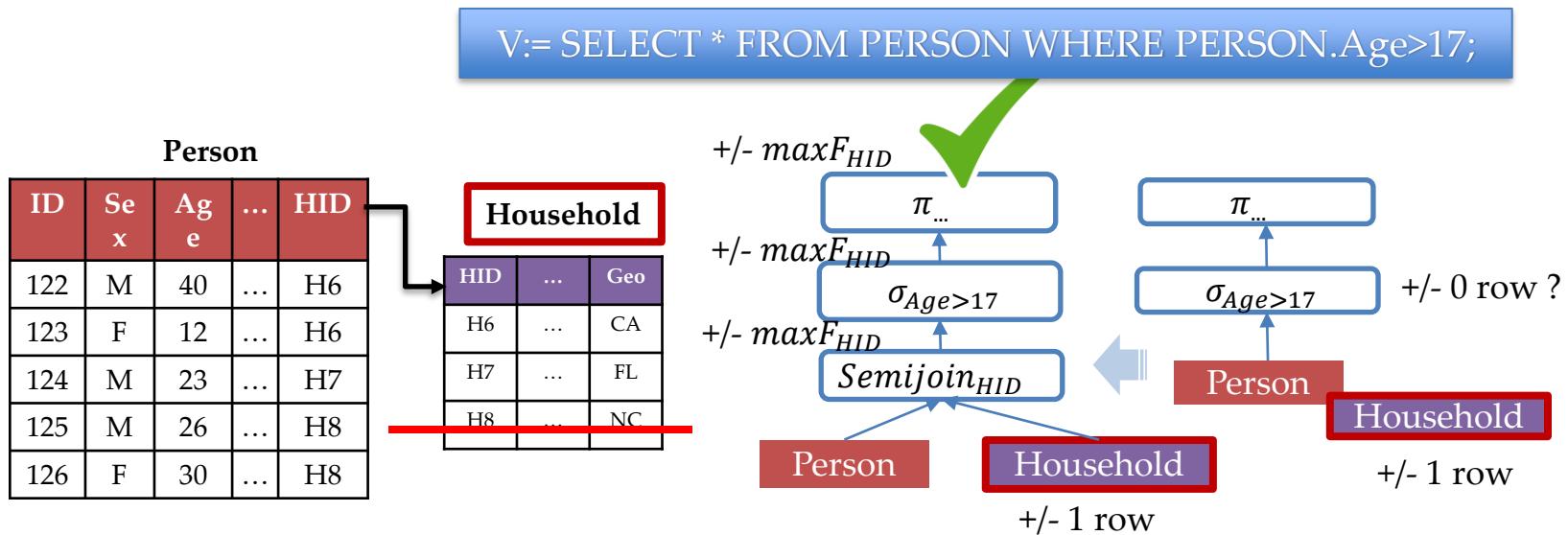
→ Household

HID	...	Geo
H6	...	CA
H7	...	FL
H8	...	NC



# Semi-join Rewrite

- Policy: Household



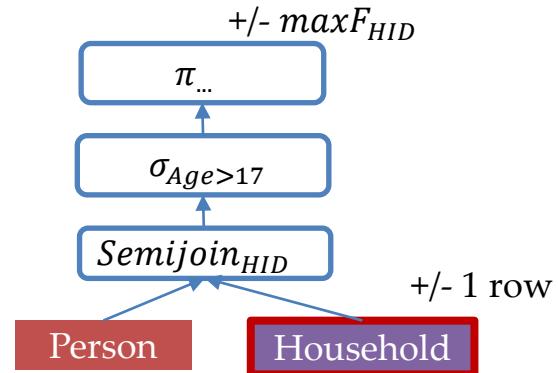
# Defining Private Objects

- Existing solutions:
  - Extend privacy policies via foreign key constraints in multi-relational DB
- Open questions:
  - Privacy data-definition language (PDDL)
    - How to define privacy with general constraints?
    - How to automatically enforce the privacy policies?



# What else could go wrong?

- High sensitivity for join query!
  - What if  $\max F_{HID}$  is not public?
- Existing solutions:

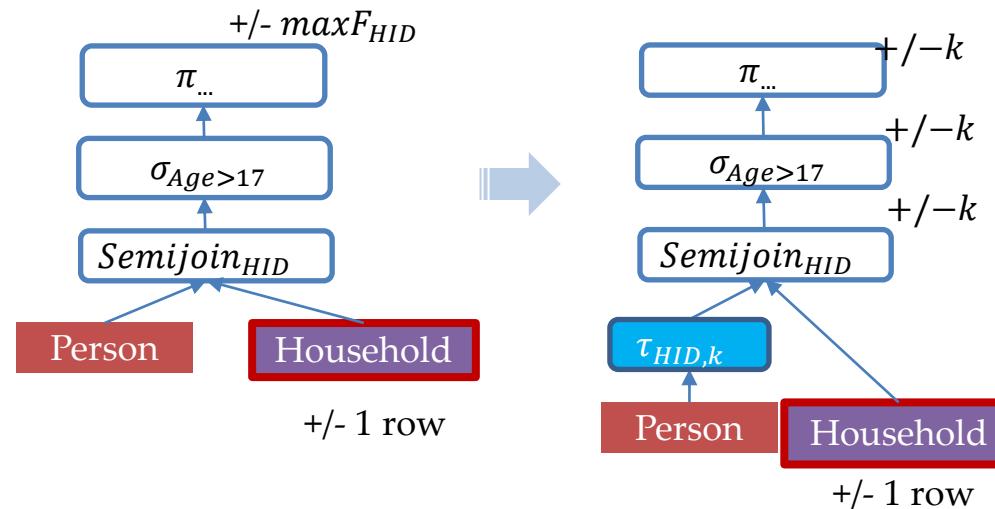


PINQ	GoogleDP	Flex	PrivateSQL
Grouping before join	Manually specified bound + sampling	Smooth sensitivity	Truncation + automatically learned thresholds; key tracking



# View Rewrite with Truncation Operator

- Add a truncation operator  $\tau_{HID,k}(\cdot)$  to bound the max multiplicity of join key



- Automatically learn the optimal  $k$  to minimize the total error



# Handling Join Queries

- Existing solutions:
  - Smooth-sensitivity, truncation/Lipchitz extension, sampling, key-tracking
- Open questions:
  - Which algorithm to use?
  - Physical implementation of DP into DBMS?
    - How to support key-tracking in the query plan?
    - How to efficiently compute tighter sensitivity upper bound for better accuracy?



# What else could go wrong?

- Handle multiple queries:
  - Unbounded privacy loss or stop query answering
- Existing solutions:
  - General DP views to answer multiple queries [KTHFMHM, VLDB19]
  - Cache historical query answers [MHRH, TPDP20]
- Open questions:
  - How to ensure consistency between queries?
  - How to allocate privacy budget among queries?
  - How to pick DP algorithms for multiple queries?



# General Implementation Issues

- Side channel attacks
  - Adversary can observe answer to their question, response time, system decision to execute the query or deny it [HPN SEC11]
- Randomness & floating-point issues
  - Laplace mechanism [Mironov CCS12], Exponential mechanism [Ilvento CCS20]



# Summary of Open Questions

- Design of Private DDL and DML
  - Protect privacy at correct resolution
  - Prevent unsafe/non-private query
  - Provide better accuracy for complex queries
- DP integration into DB
  - Add on top of existing DB systems
  - Reimplement DP components (e.g. query plan, query rewriting, key tracking, cache/synthetic data) for better utility and/or performance



# **MODULE 3**

# **SYSTEM INTEGRATION &**

# **OPTIMIZATION**

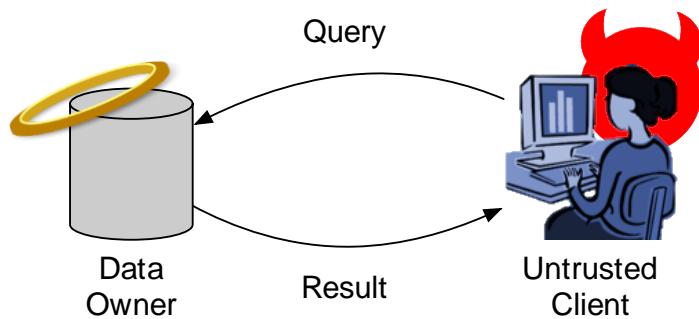
# Overview

- Part 1: DBMS Security and Privacy
- Part 2: State-of-the-Art Solutions
- Part 3: Open Challenges

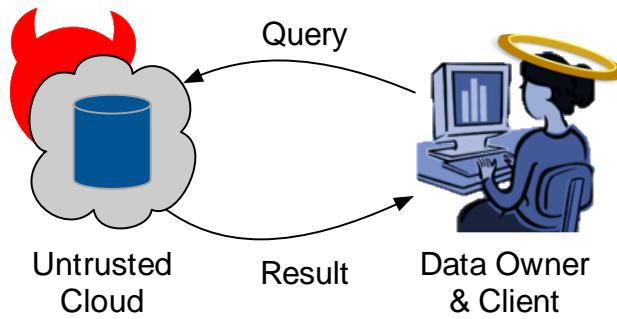
# Part 1: DBMS Security and Privacy

# Security and Privacy (S&P) Settings

## Client/Server



## Untrusted Cloud

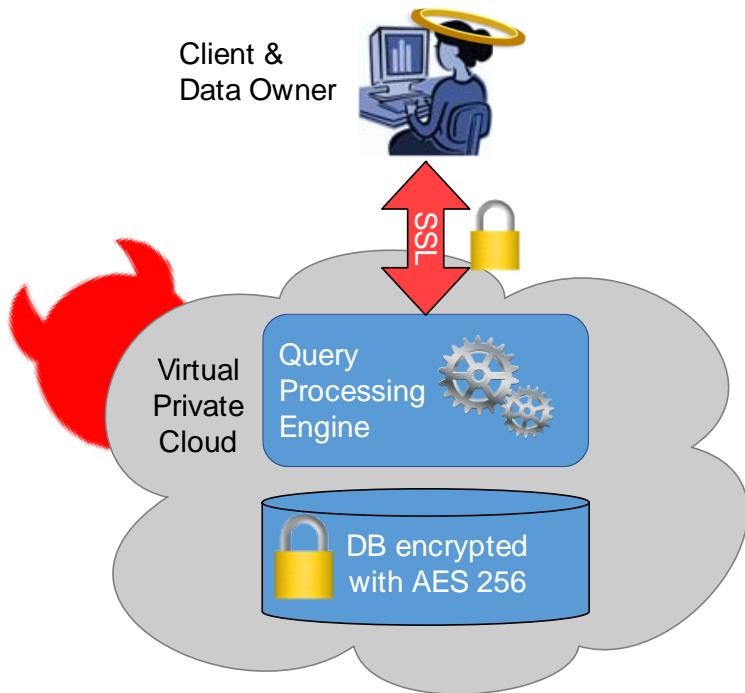


- Column-at-a-time access control policy
- Systems behave exactly like a standard DBMS from client's perspective

# Why can't I just add a password to my DBMS? And use encrypted storage?

- Your attack surface is the entire stack
- Your data may reside on untrusted cloud servers
- We need to re-architect our systems with security and privacy at the forefront

# Naïve DBMS deployment on an untrusted cloud



## What could go wrong?

- Storage: National Security Letter compels service provider to decrypt data
- Query processing: insider threat sees data-dependent query traces and result sizes
- Client side: rogue user systematically queries DB to deduce its private contents

Regulatory compliance ≠ meaningful S&P guarantees!

# What about existing work from the security and privacy community?

- Existing S&P solutions are piecemeal – they address specific steps in the DBMS workflow
  - MPC & TEE: Protects data *during computation*
  - DP: Protects data when *releasing results*
- Composing these techniques is non-trivial

# Where do we come in?

- To date we've mostly focused on making the DBMS fast and scalable – to great success!
- S&P is usually an afterthought
- We have a lot to offer in this emerging space of making privacy-preserving analytics practical and usable

## End-to-end S&P guarantees

- Covers from when a client submits a query to when they receive their results

## Efficient

- Offers performance comparable to that of current DBMSs

## Robust

- Supports ad-hoc query workloads and diverse user needs

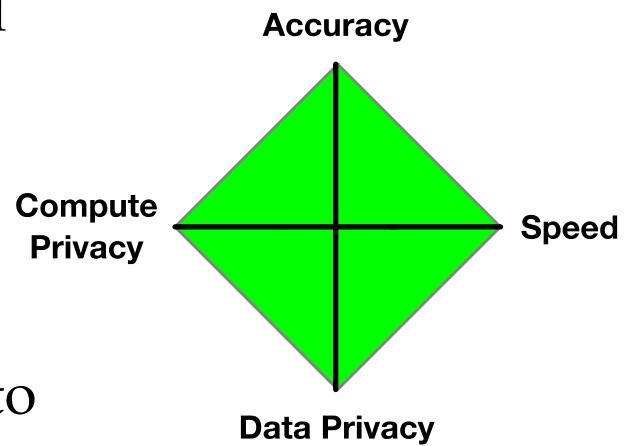
## Usable

- UI like that of a standard DBMS for low barrier to entry. Understandable S&P guarantees.

# DBMS Goals

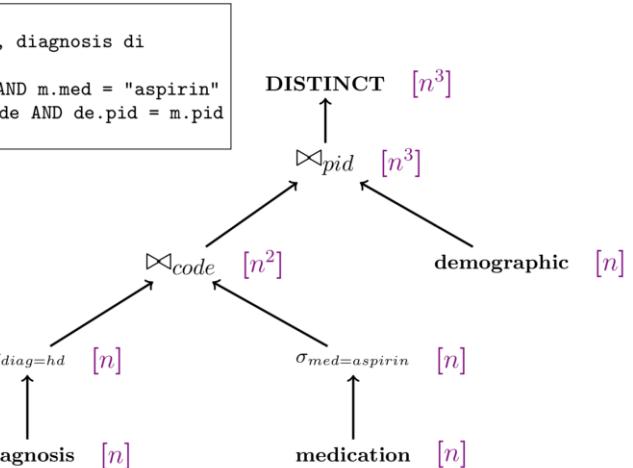
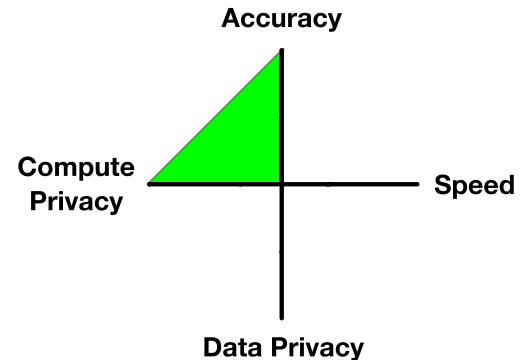
- Rigorous, explainable guarantees of:
  - Privacy of storage and computation over data
  - Privacy of data itself, esp. under repeated querying
- Maximize:
  - Query result accuracy
  - The speed of query execution

S&P guarantees are not boolean, this leads to interesting trade-offs



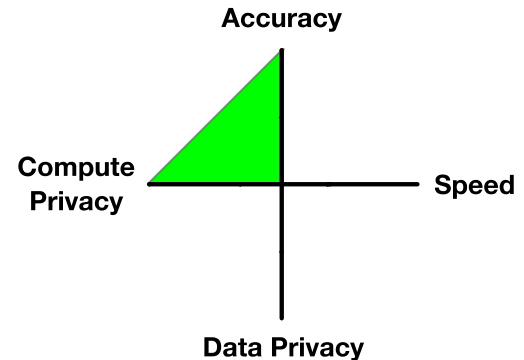
# Approach #1: Secure Computation

- MPC & TEEs protect data during query evaluation with:
  - Data encrypted in flight
  - Oblivious, data-independent execution transcript
- Useful for query processing in untrusted cloud
- *MPC is really slow! 1,000X+ slower than running in the clear*
- *TEEs require specialized hardware and depends on chip vendors to have correct implementations*

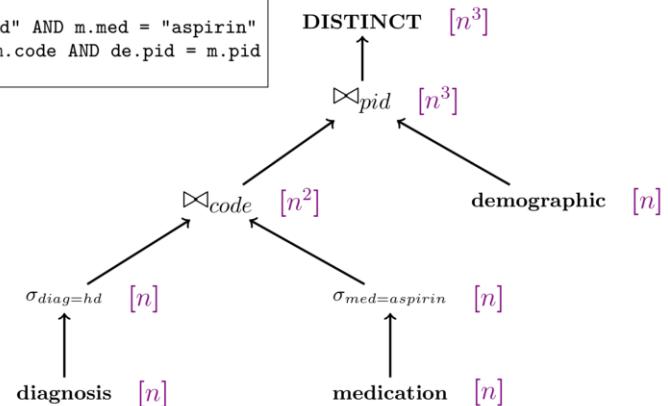


# Approach #1: Secure Computation

- MPC & TEEs do not protect data during data release
- Repeated queries of a MPC/TEE-only system leaks the private data distribution
- *Revealing any data-dependent computation, such as intermediate result sizes, leaks private information*

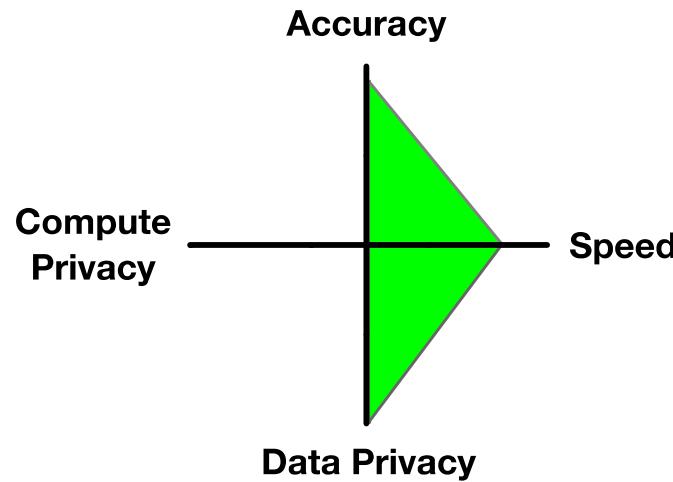


```
SELECT DISTINCT pid
FROM demographic de, diagnosis di
    medication m
WHERE di.diag="hd" AND m.med = "aspirin"
    AND di.code = m.code AND de.pid = m.pid
```



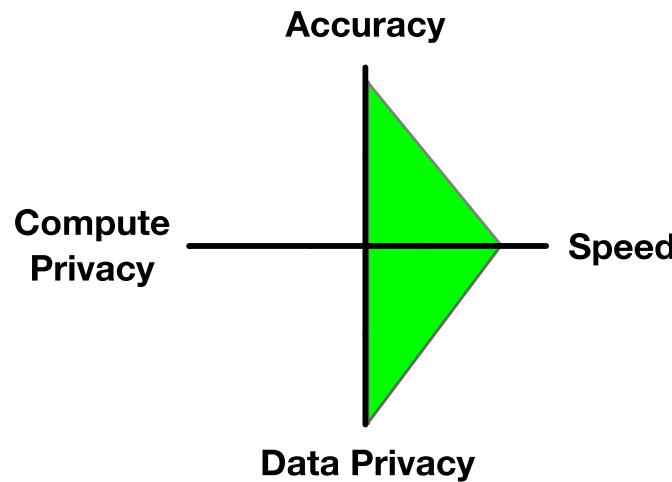
# Approach #2: DP

- Differential privacy reveals statistics about DB records while withholding info about individual input tuples [Dwork06]
- Injects precisely calibrated levels of noise into query answers proportional to individual contributions
- It is composable.
- *Cumulative information leakage subject to a privacy budget,  $\epsilon$*



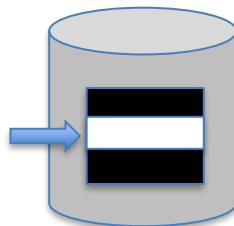
# Approach #2 : DP

- DP-only systems add noise to either the query result computed by the server or the input data from data owners
- Adding noise to the query result *requires a single data owner* that serves as a trusted server
- When considering multiple data owners, applying DP to input data adds *error proportional to the number of owners*

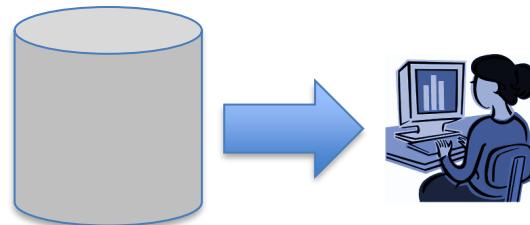


# Why can't we naively integrate these building blocks into existing systems?

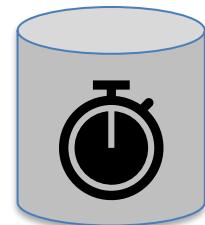
## Problem 1: Preventing Side Channel Attacks



Access Pattern Leakage



Volume Leakage



Timing Leakage

# Why can't we naively integrate these building blocks into existing systems?

## Problem 1: Preventing Side Channel Attacks

### Access Pattern Leakage

Memory locations accessed during computation

→ Leaks frequency of values in source data

### Volume Leakage

Size of computed results

→ Leaks number of records processed at each operator

### Timing Leakage

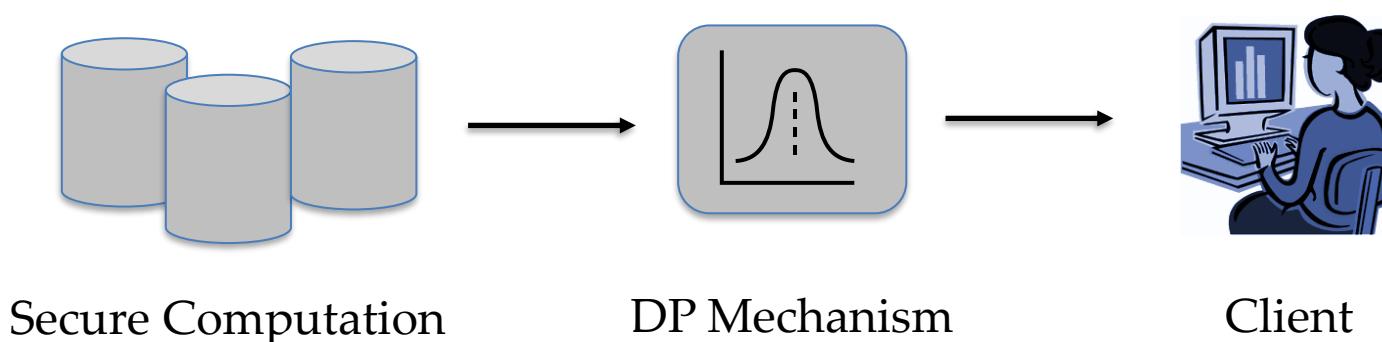
Time required for computation

→ Leaks time needed to process output of each operator

**Side channel leakage reveals distribution of values in private source data**

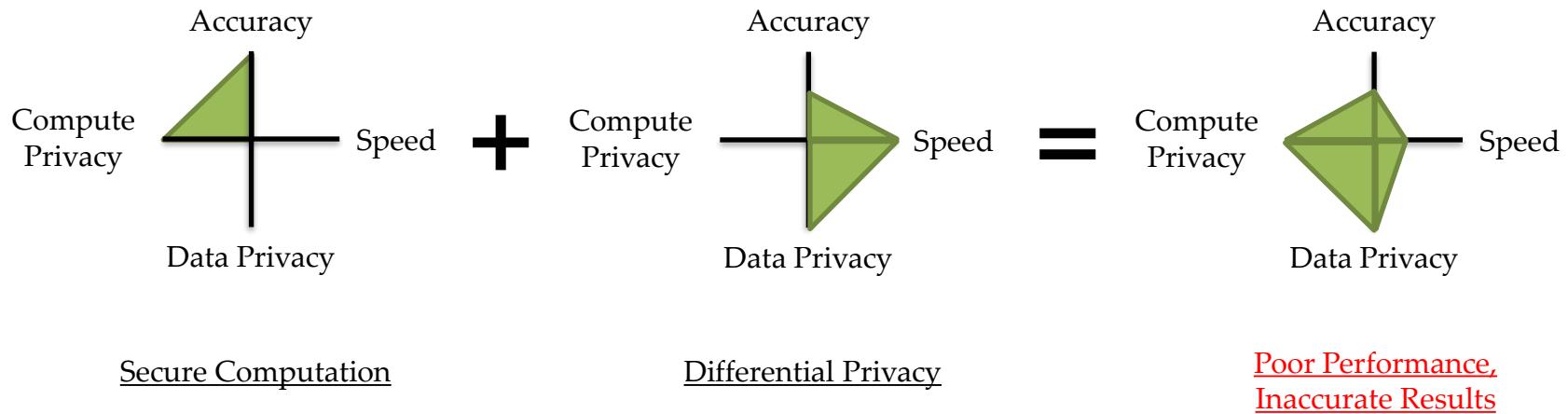
Why can't we naively  
integrate these building  
blocks into existing  
systems?

Problem 2: Properly Composing Techniques



# Why can't we naively integrate these building blocks into existing systems?

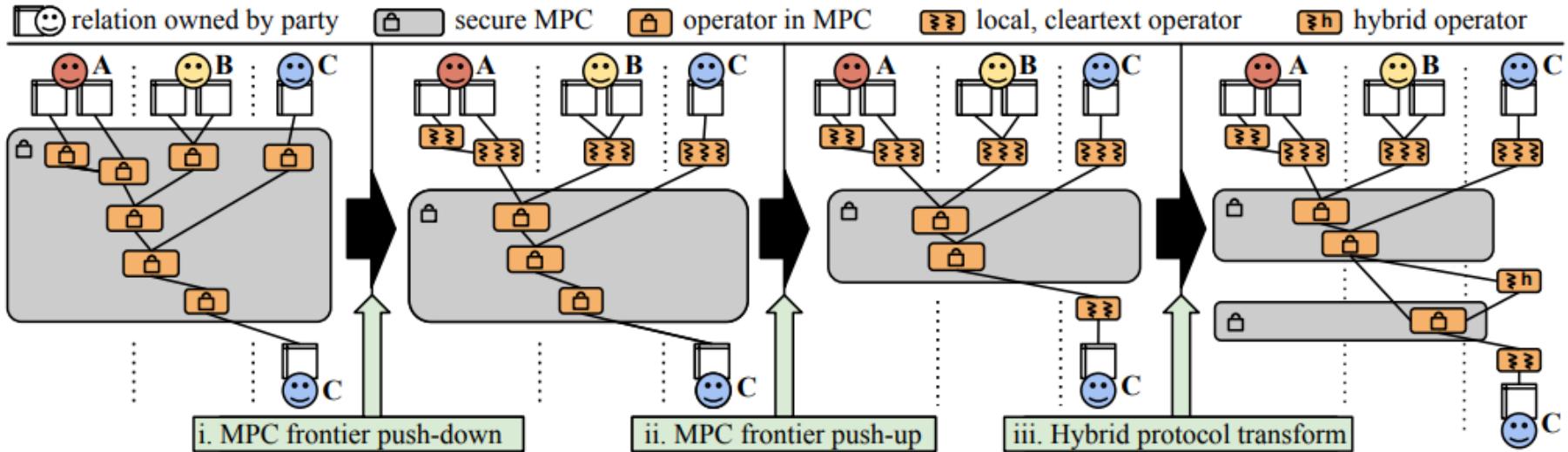
## Problem 2: Properly Composing Techniques



## Part 2: State-of-the-Art Solutions

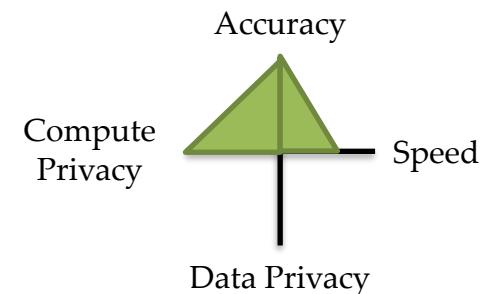
# Conclave: Secure Multi-Party Computation on Big Data

- Compiler for relation queries that accelerates secure computation
- Instead of directly converting SQL queries into secure computation, transforms queries into a combination of data-parallel, local cleartext processing and small MPC steps



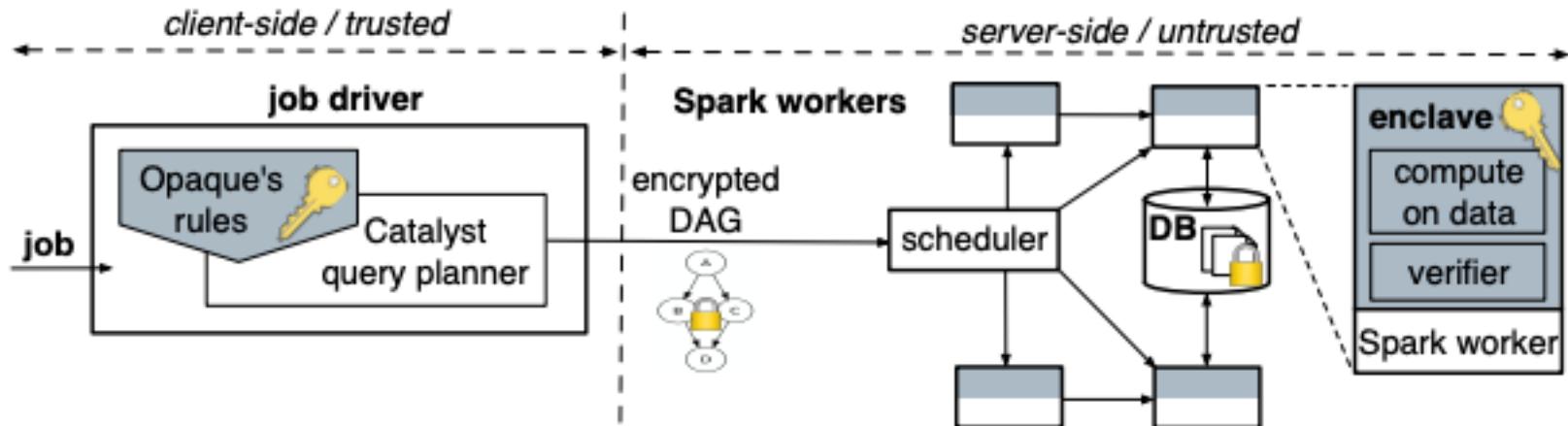
# Conclave: Secure Multi-Party Computation on Big Data

- Compiler for relation queries that accelerates secure computation
- Instead of directly converting SQL queries into secure computation, transforms queries into a combination of data-parallel, local cleartext processing and small MPC steps
- Prevents side-channel attacks
- Improves speed through less MPC
- Still requires some expensive MPC
- Only protects data during computation



# Opaque: An Oblivious and Encrypted Distributed Analytics Platform

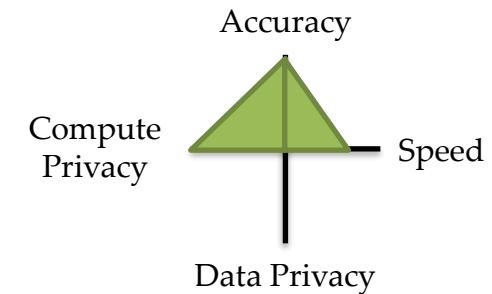
- Distributed analytics platform built on Spark that utilizes TEEs for oblivious execution
- Introduces oblivious relational operators for rule and cost-based query optimization with security and privacy guarantees



Wenting Zheng, Ankur Dave, Jethro G. Beekman, Raluca Ada Popa, Joseph E. Gonzalez, and Ion Stoica. Opaque: An oblivious and encrypted distributed analytics platform. *USENIX 2017*.

# Opaque: An Oblivious and Encrypted Distributed Analytics Platform

- Distributed analytics platform built on Spark that utilizes TEEs for oblivious execution
- Introduces oblivious relational operators for rule and cost-based query optimization with security and privacy guarantees
- Prevents side-channel attacks
- Improves speed through TEEs
- Requires specialized hardware
- Only protects data during computation



Wenting Zheng, Ankur Dave, Jethro G. Beekman, Raluca Ada Popa, Joseph E. Gonzalez, and Ion Stoica. Opaque: An oblivious and encrypted distributed analytics platform. *USENIX 2017*.

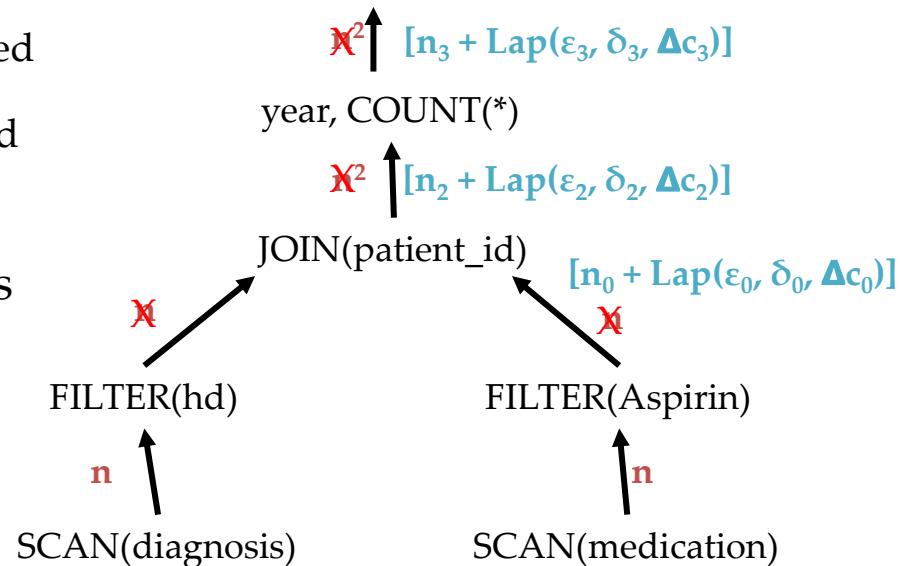
# Shrinkwrap: Secure Query Execution with DP Guarantees

Extended S&P guarantees with DP to cover:

- Prior knowledge of data owners wrt query answers
- Cumulative privacy loss from repeated querying
- Collusion among the data owners and the client

Reveal noisy intermediate cardinalities at runtime for speedup

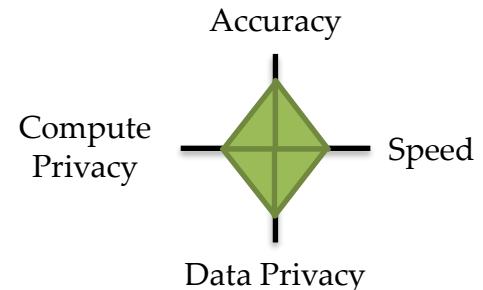
Noise query answers in MPC, true result revealed to no one



# Shrinkwrap: Secure Query Execution with DP Guarantees

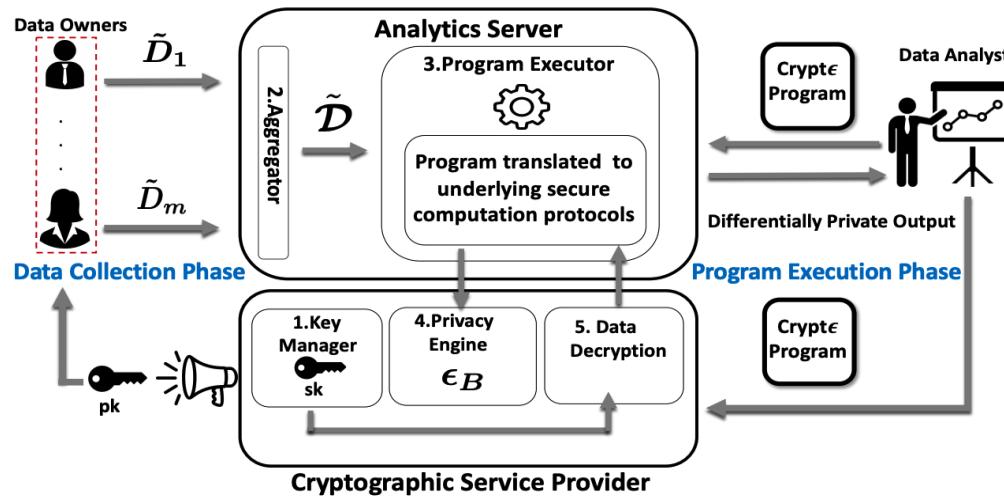
Extended S&P guarantees with DP to cover:

- Prior knowledge of data owners wrt query answers
  - Cumulative privacy loss from repeated querying
  - Collusion among the data owners and the client
- 
- Prevents side-channel attacks
  - Improves speed through DP volume
  - Requires user-defined trade-offs
  - Limits on repeated querying



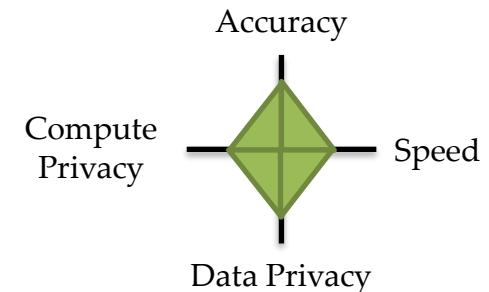
# CRYPT $\epsilon$

- Crypto-assisted system that combines MPC with DP to execute database queries over data collected from multiple data owners
- Achieves strong accuracy guarantees without requiring data owners to upload private data to a central trusted data collector



# CRYPT $\epsilon$

- Crypto-assisted system that combines MPC with DP to execute database queries over data collected from multiple data owners
- Achieves strong accuracy guarantees without requiring data owners to upload private data to a central trusted data collector
- Ensures privacy through DP
- Improves accuracy through MPC
- Requires expensive MPC computation
- Only supports linear, aggregate queries

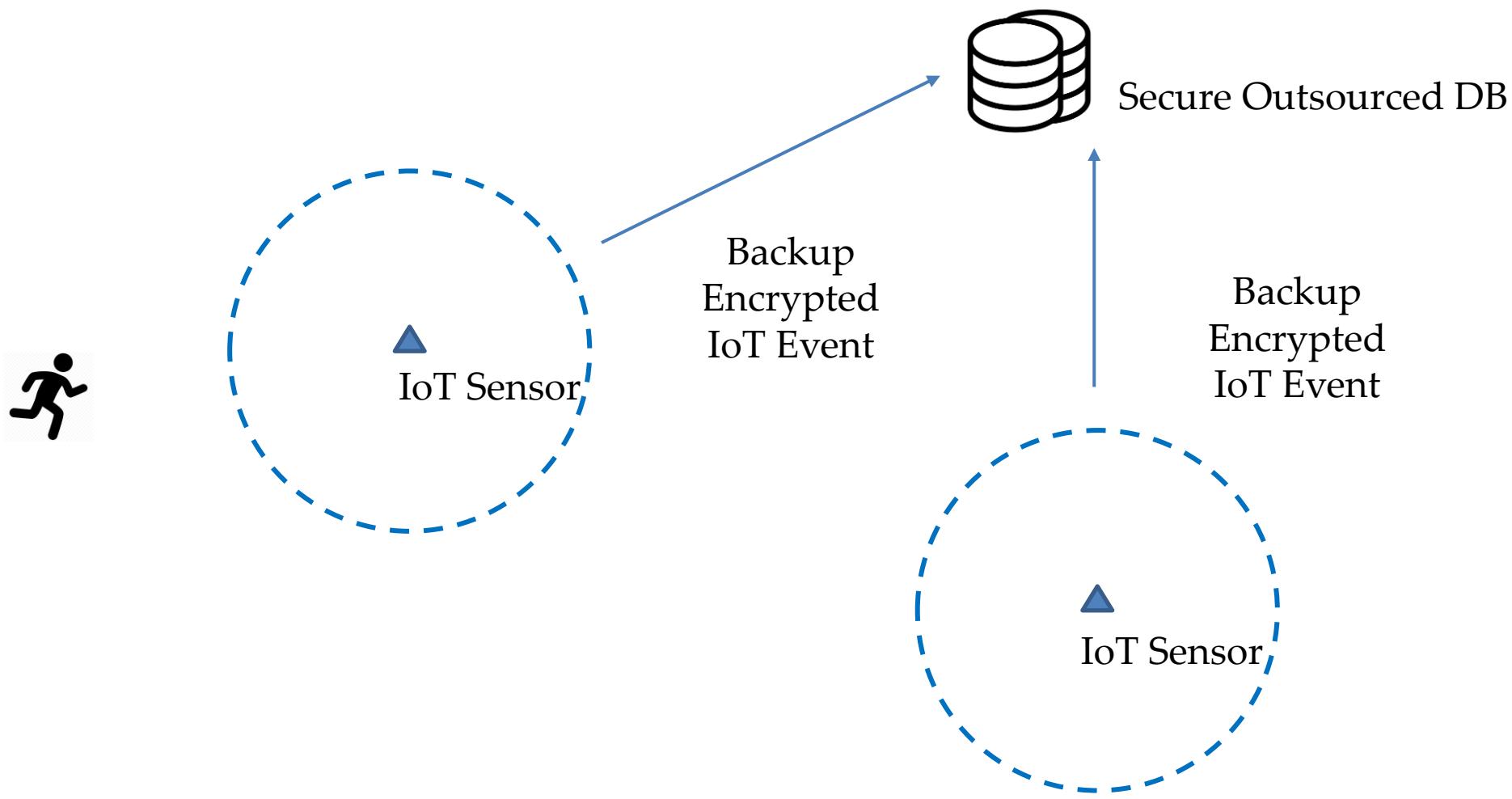


# Part 3: Open Challenges

# Growing Databases

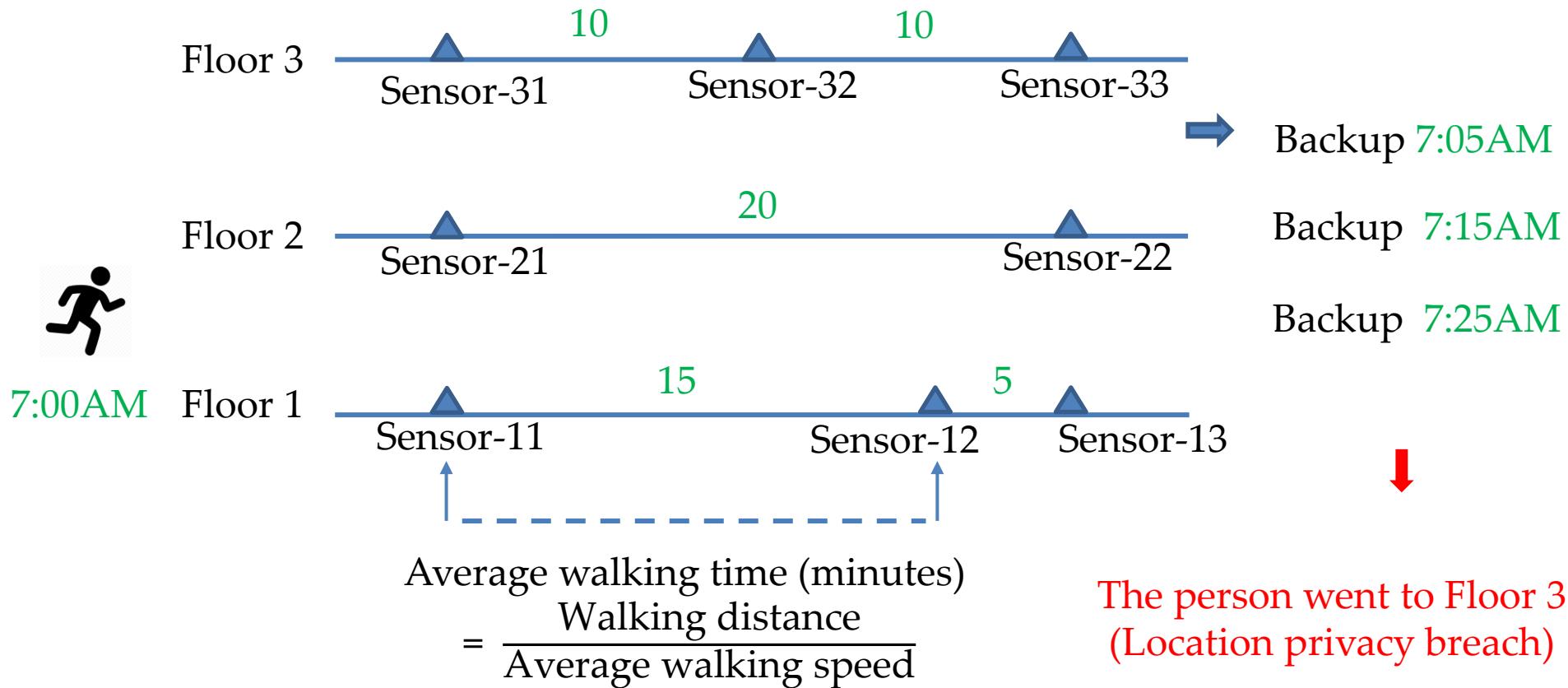
- Additional leakages: Update pattern.
- Revealing exact update pattern could lead to privacy breach.
- Require further countermeasures that hides update pattern.

# Real-world Privacy Breach



IoT Sensor: WiFi access point, smart light bulb, population counter, etc.

# Real-world Privacy Breach



# Real-world Privacy Breach

- The event time is strongly tied with the backup time (database update time).
- An adversary can breach privacy by using the timing information of updates.
- This type of attack generalizes to any event-driven update where the event time is tied to the data upload time.
- SIGMOD21 paper: DP-Sync

# Transactions

- Additional requirements (ACID)

# Query Interfaces

- Additional user parameters (e.g., privacy budget, algorithm selection)
- Requires deep DP and MPC knowledge on the part of clients

# Floating Point Support

- With MPC, doable but slow
- With DP, many attacks on floating point implementations