Building Privacy-Aware Database Systems

CS848 Winter 2021

Module 2

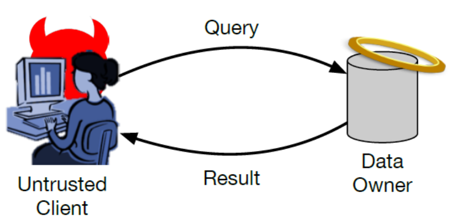
UNIVERSITY OF WATERLOO

DSg Data Systems Group

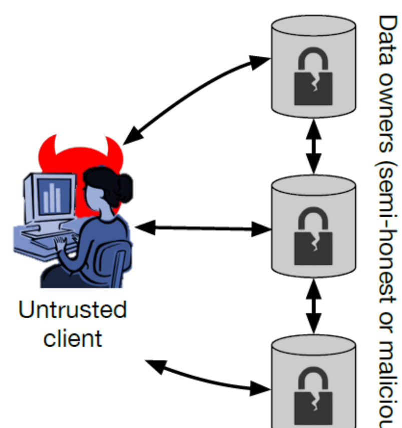
This course will explore …

- How to define a good privacy promise?
- How to design a privacy-preserving algorithms?
- How to build a privacy-aware database systems?

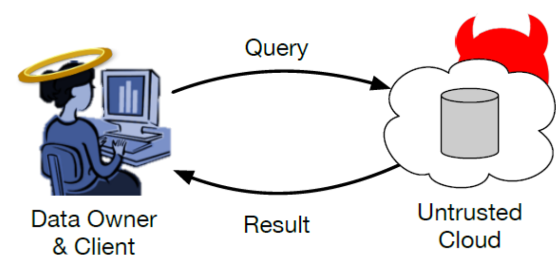
Greatly depend on
the architecture setup and trust assumptions



Client-server with
trusted data curator



Data federation



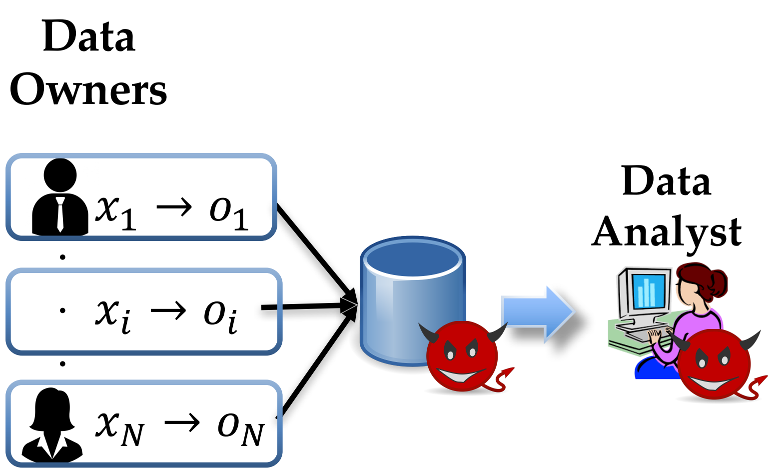
Cloud service provider

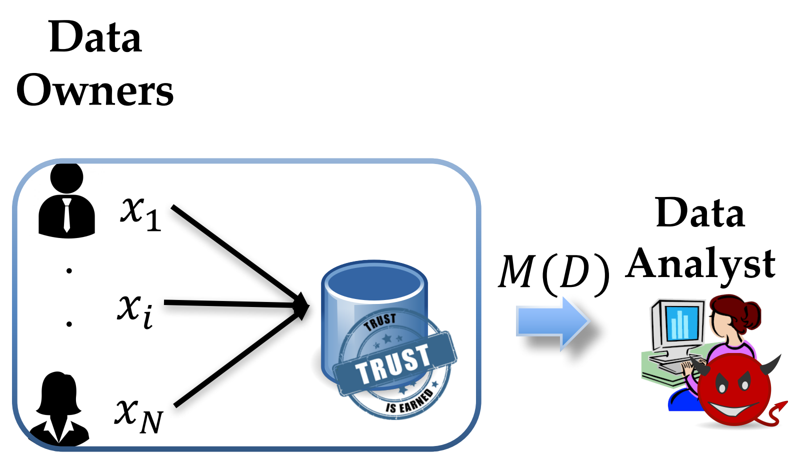
Outline

- Part I: Local Differential Privacy (LDP)
- Part II: Marrying DP with Crypto
- Upcoming Papers and Announcements

No Trusted Data Curator

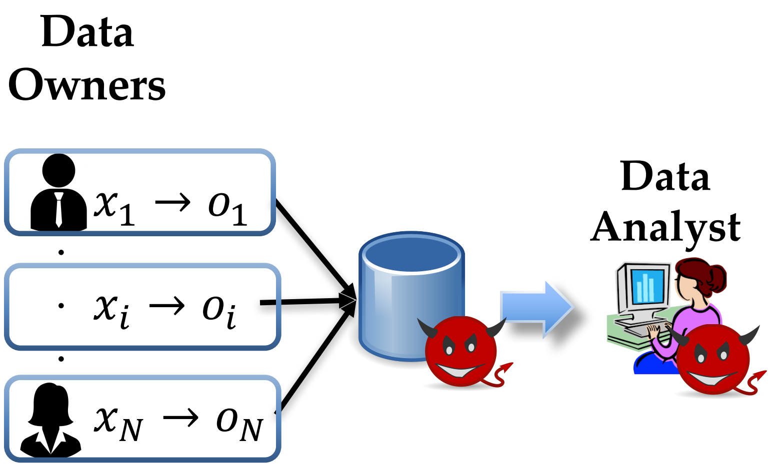
- Local DP
  - No trusted data curator
- Centralized DP
  - Trusted data curator

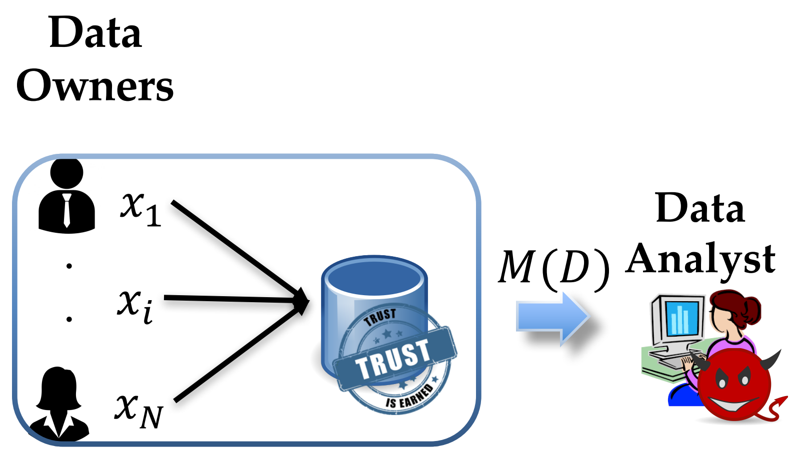




No Trusted Data Curator

- Local DP
  - No trusted data curator
- Centralized DP
  - Trusted data curator





$$\ln\left(\frac{\Pr[A(x_i) = o]}{\Pr[A(x_i{}') = o]}\right) \leq \varepsilon$$

$$\ln\left(\frac{\Pr[M(D) = o]}{\Pr[M(D') = o]}\right) \leq \varepsilon$$

[W 65]

Randomized Response (a.k.a. local randomization)

D

Disease (Y/N)
Y
Y
N
Y
N
N

With probability p,
Report true value

With probability 1-p,
Report flipped value

O

Disease (Y/N)
Y
N
N
N
Y
N

Privacy Analysis of RR

- Considering a record taking values (x, x′)
- Consider some output/response O

		Output	
		Yes	No
Input	Y	p	1-p
	N	1-p	p

$$e^{-\epsilon} \leq \frac{\Pr[A(N) = No]}{\Pr[A(Y) = No]} \leq e^{\epsilon}$$

$$e^{-\epsilon} \leq \frac{\Pr[A(Y) = Yes]}{\Pr[A(N) = Yes]} \leq e^{\epsilon}$$

$$\frac{1}{1+e^{\epsilon}} \leq p \leq \frac{e^{\epsilon}}{1+e^{\epsilon}} \Leftarrow e^{-\epsilon} \leq \frac{p}{1-p} \leq e^{\epsilon}$$

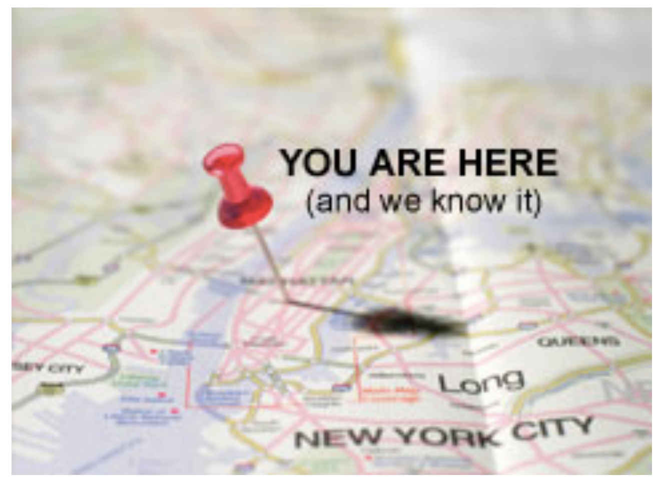
Utility Analysis of RR

- Suppose y out of N people replied "yes", and rest said "no"; what is the best estimate for π = fraction of people with disease = Y?
- Expected number of "yes" responses:
$$E[y] = \pi N \cdot p + (1-\pi)N \cdot (1-p)$$
- Unbiased estimator for π: $\hat{\pi} = \dfrac{\frac{y}{N} - (1-p)}{2p-1}$
  - $E(\widehat{\pi}) = E\left[\dfrac{\frac{y}{N}-(1-p)}{2p-1}\right] = \pi$
  - $Var(\hat{\pi}) = \dfrac{\pi(1-\pi)}{N} + \dfrac{1}{N\left(16\left(p-\frac{1}{2}\right)^2 - \frac{1}{4}\right)}$

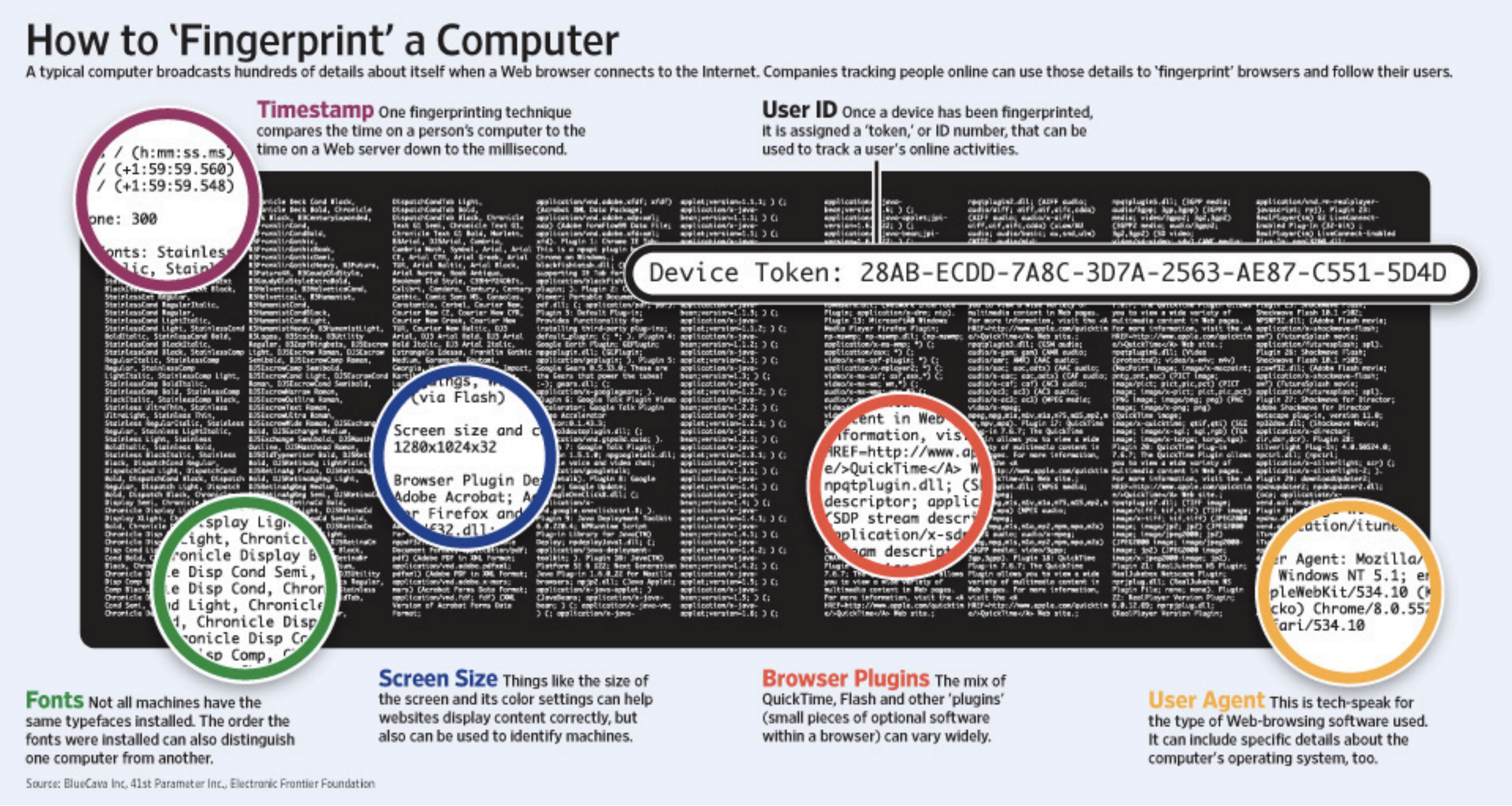
Sampling **Variance due to coin flips**

RR for Larger Domains (mini-assignment 2)

- Suppose area is divided into k x k uniform grid.
- How to achieve LDP?
  - What is the probability of reporting the true location?
  - What is the probability of reporting a false location?

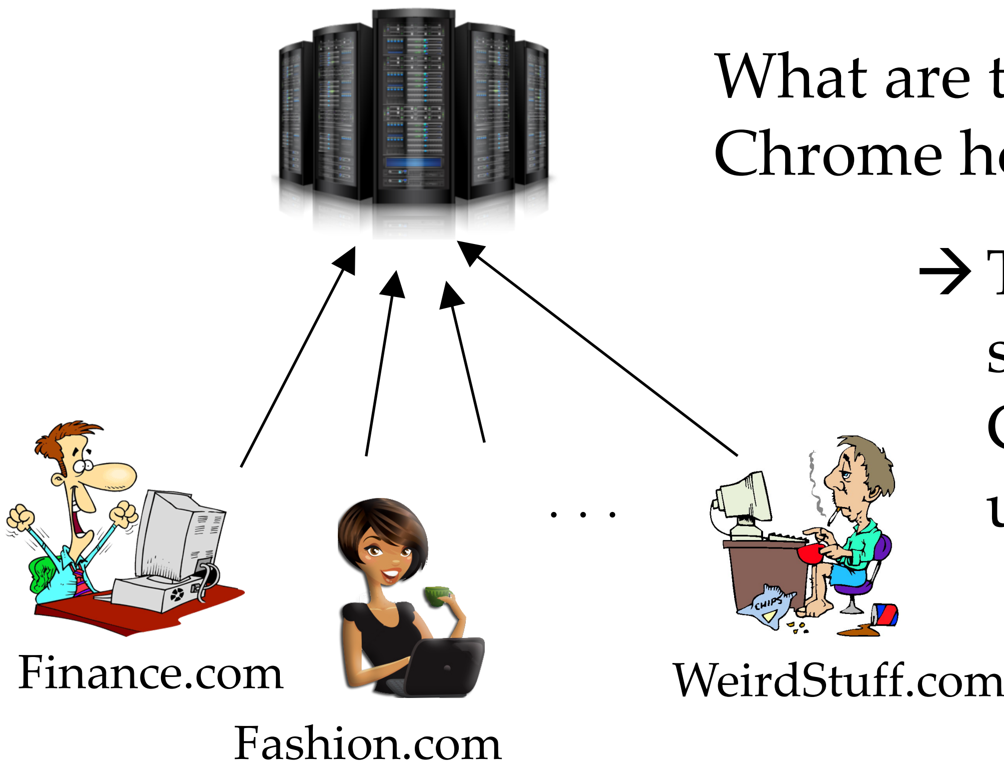


Browser configurations can identify users



Problem

[Erlingsson et al CCS'14]



What are the *frequent* unexpected Chrome homepage domains?

→ To learn malicious software that change Chrome setting without users' consent

How to ensure privacy?

Can use Randomized Response …

On a binary domain:
  With probability p report true value
  With probability 1-p report false value

… but the domain of all urls is very large …
… original value is reported with very low prob.

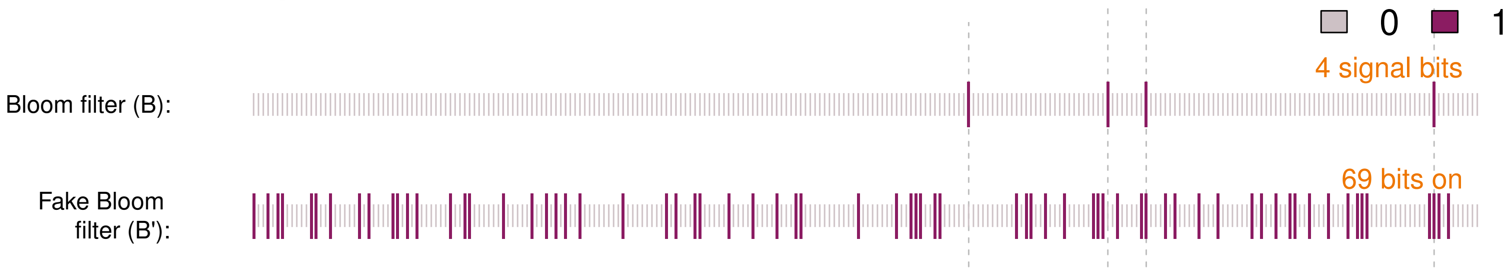
RAPPOR Solution

- Idea 1: Use bloom filters to reduce the domain size

Finance.com

RAPPOR Solution

- Idea 2: Use RR on bloom filter bits



RAPPOR Solution

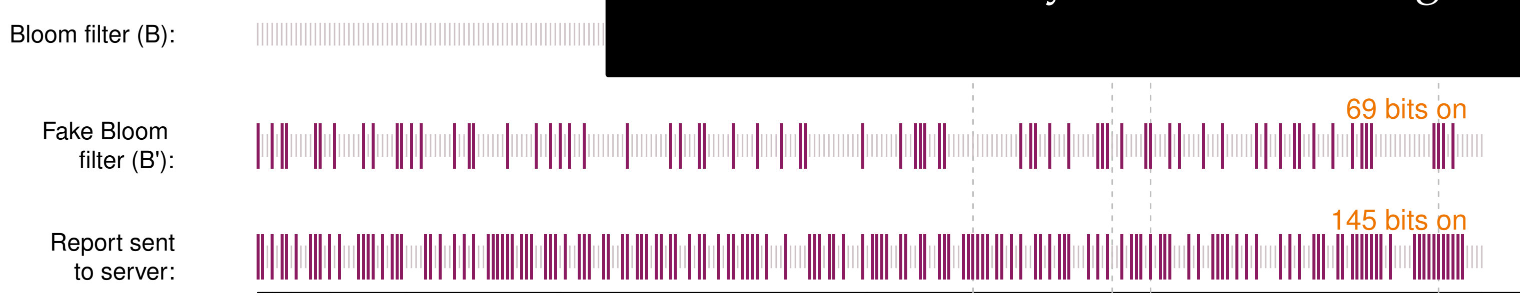
- Idea 3: Again use RR on the Fake bloom filter

Finance.com

Why randomize two times?

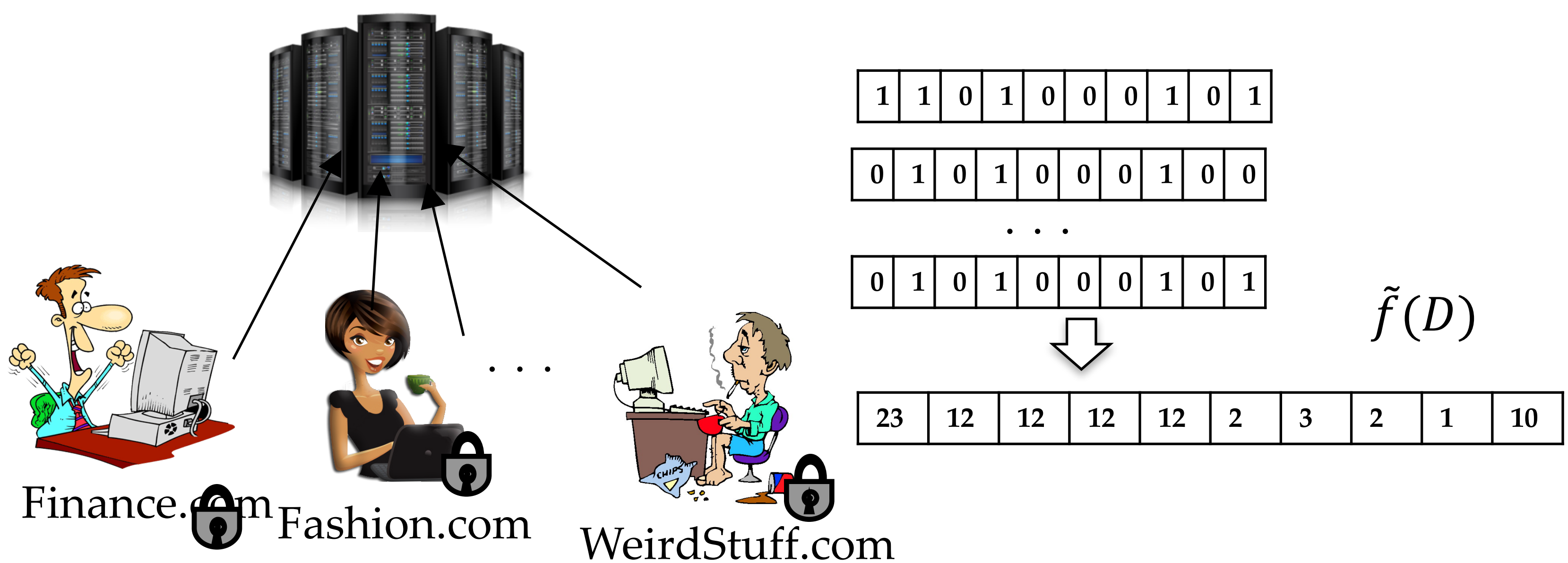
- Chrome collects information each day

- Want perturbed values to look different on different days to avoid linking



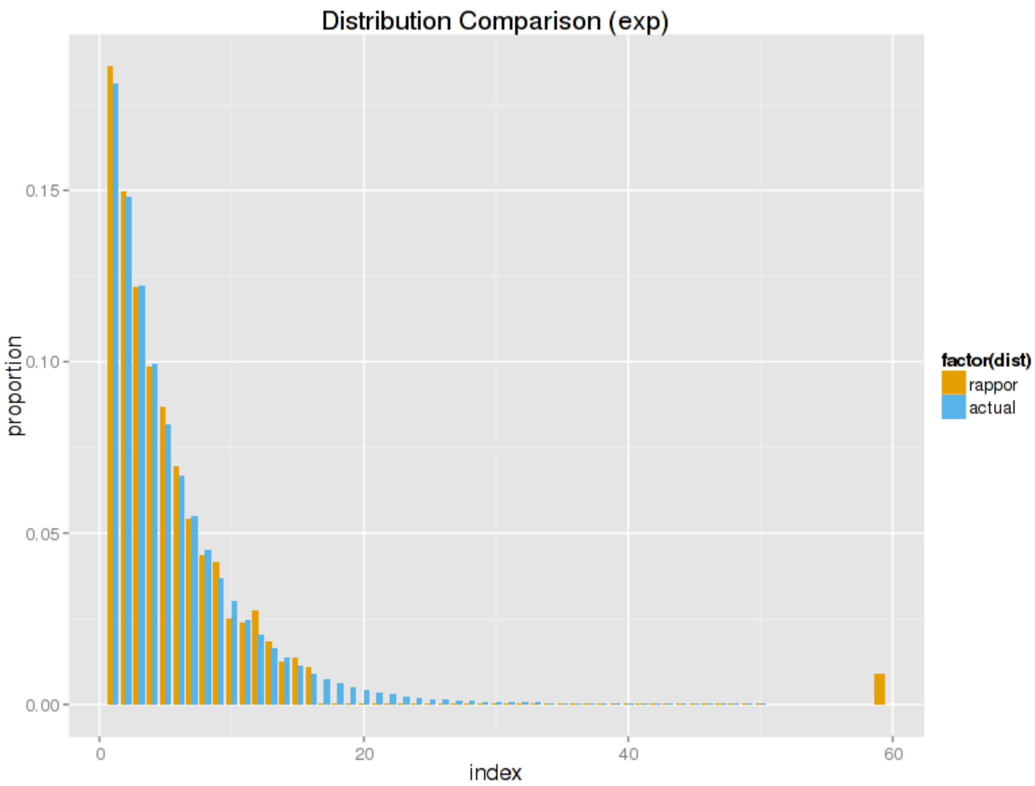
Server Report Decoding

- Step 4: estimates bit frequency from reports $\tilde{f}(D)$
- Step 5: estimate frequency of candidate strings with regression from $\tilde{f}(D)$



Evaluation

http://google.github.io/rappor/examples/report.html



Simulation Input

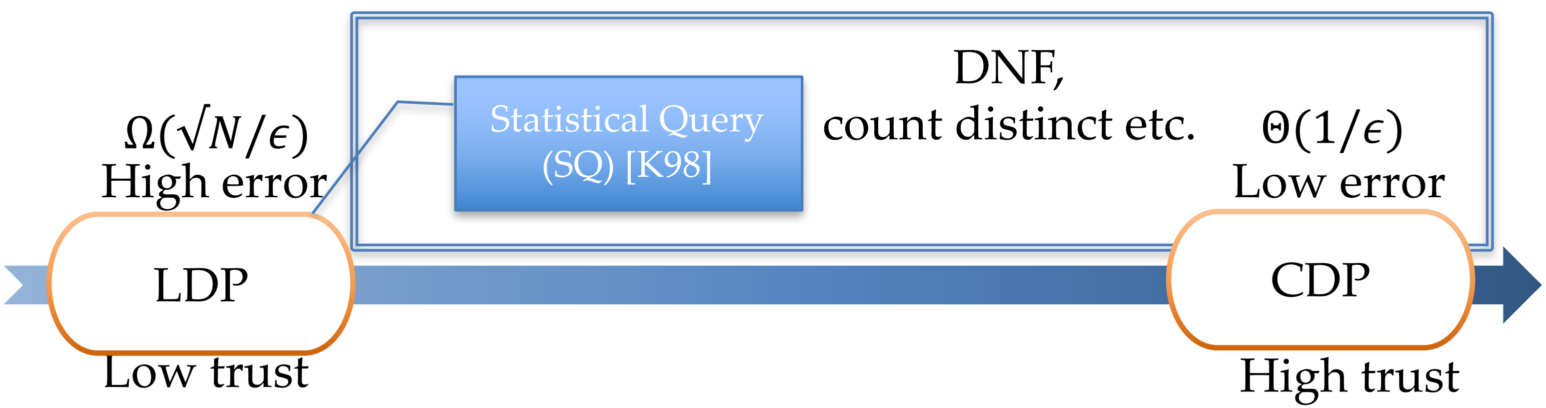
Number of clients	100,000
Total values reported / obfuscated	700,000
Unique values reported / obfuscated	50

RAPPOR Parameters

k	Size of Bloom filter in bits	16
h	Hash functions in Bloom filter	2
m	Number of Cohorts	64
p	Probability p	0.5
q	Probability q	0.75
f	Probability f	0.5

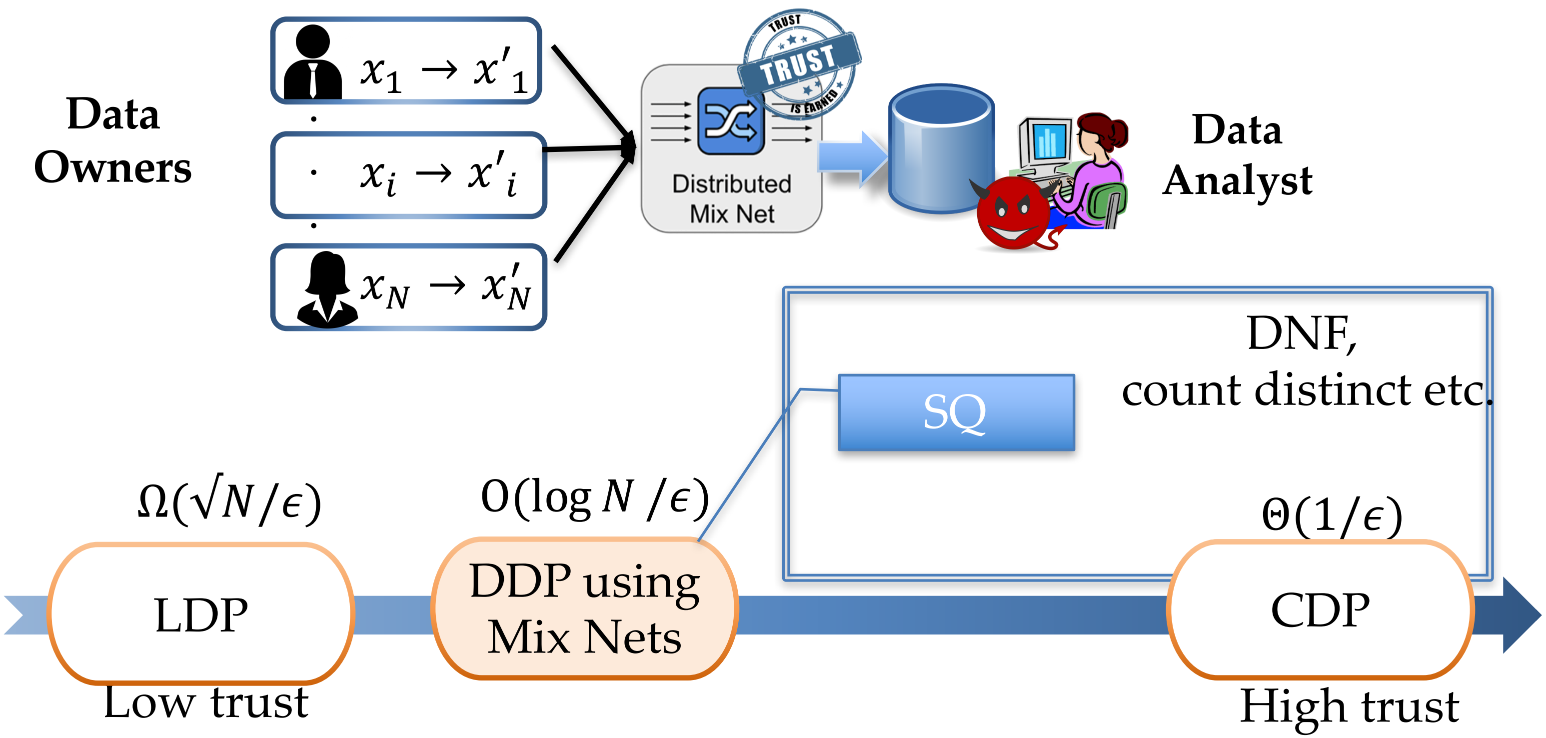
Limitations of Local DP

- Local DP: Less accurate/expressive
  - $\Omega(\sqrt{N}/\epsilon)$ for statistical counting queries, where N is datasize
  - Separation results between the accuracy and sample complexity of LDP and CDP [KLNRS08]
    - E.g. disjunctive normal form (DNF) queries



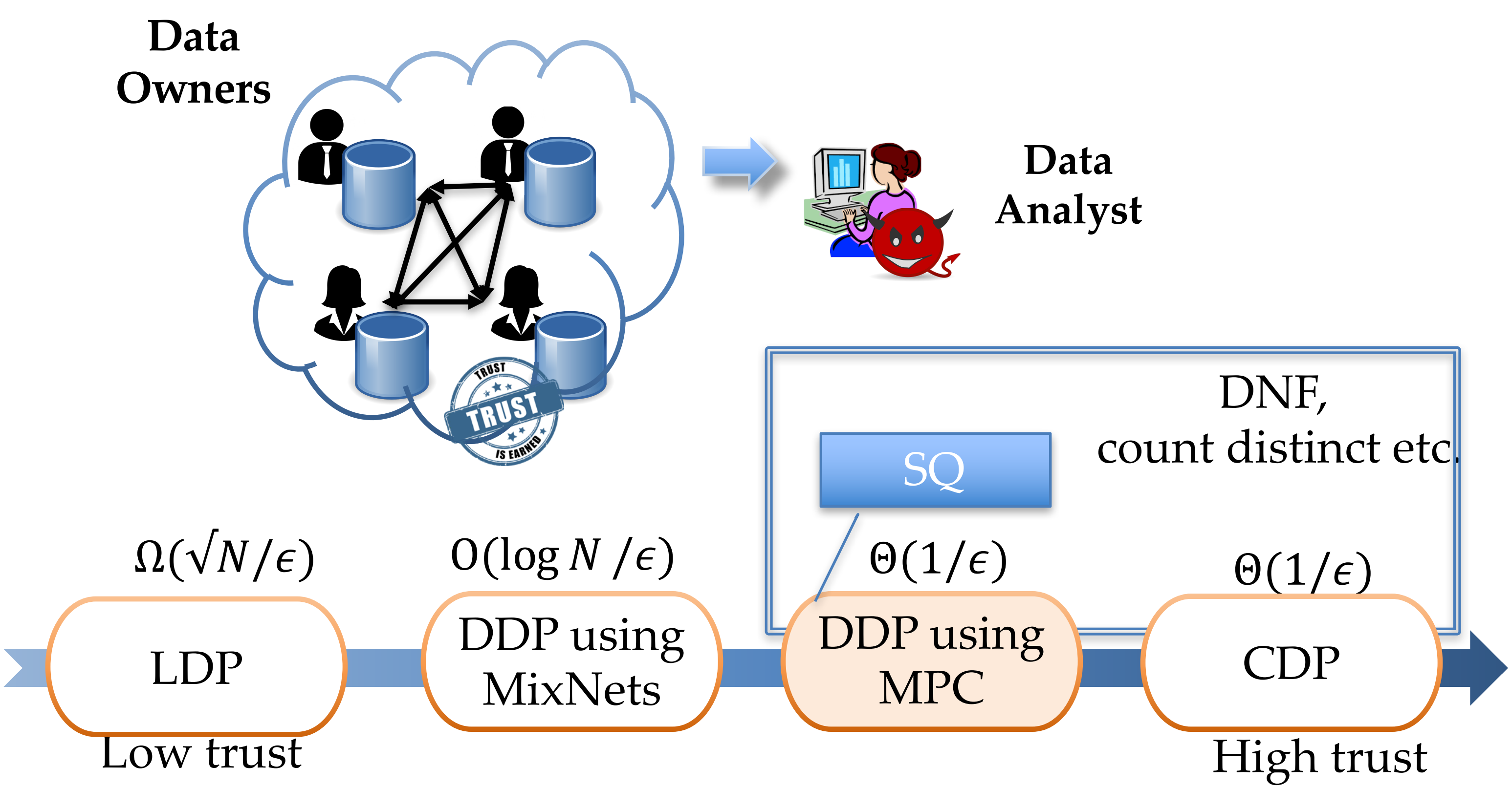
Shifting Trust Assumptions

- Trusted anonymous communication channels [BEMMRLRKTS17, CSUZZ18, EFMRTT19, BBGN19]



Shifting Trust Assumptions

- Trusted multi-party secure computation (MPC)
  [NH12, BEEGKR17, AHKM18]

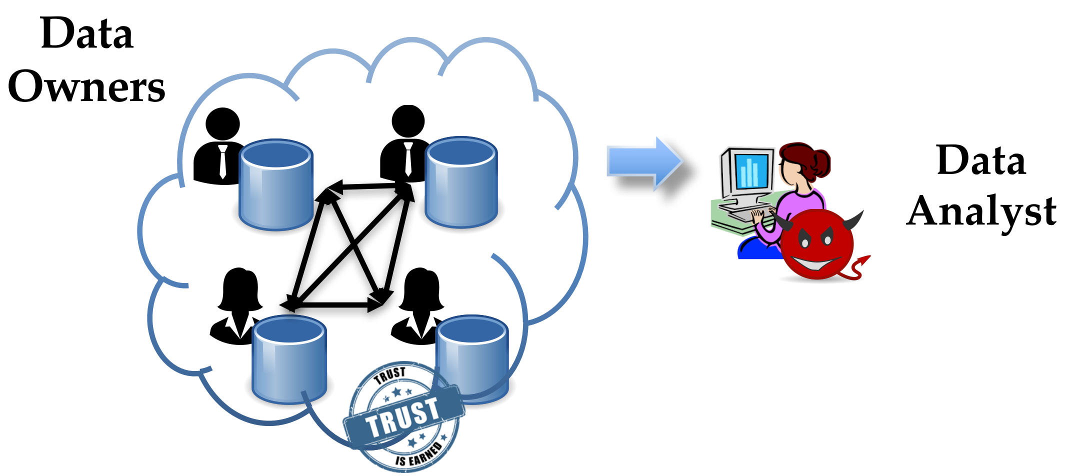


Outline

- Part I: Local DP
  - Randomized Response
  - RAPPOR
  - Limitations of Local DP
- Part II: Marrying DP with Crypto
  - Secure Multi-party computation
  - Crypte
- Upcoming Papers and Annoucements

No Trusted Data Curator

- Trusted multi-party secure computation (MPC) [NH12, BEEGKR17, AHKM18]



- MPC: (informally) to compute a function of private inputs without revealing information about the inputs beyond what is revealed by the function

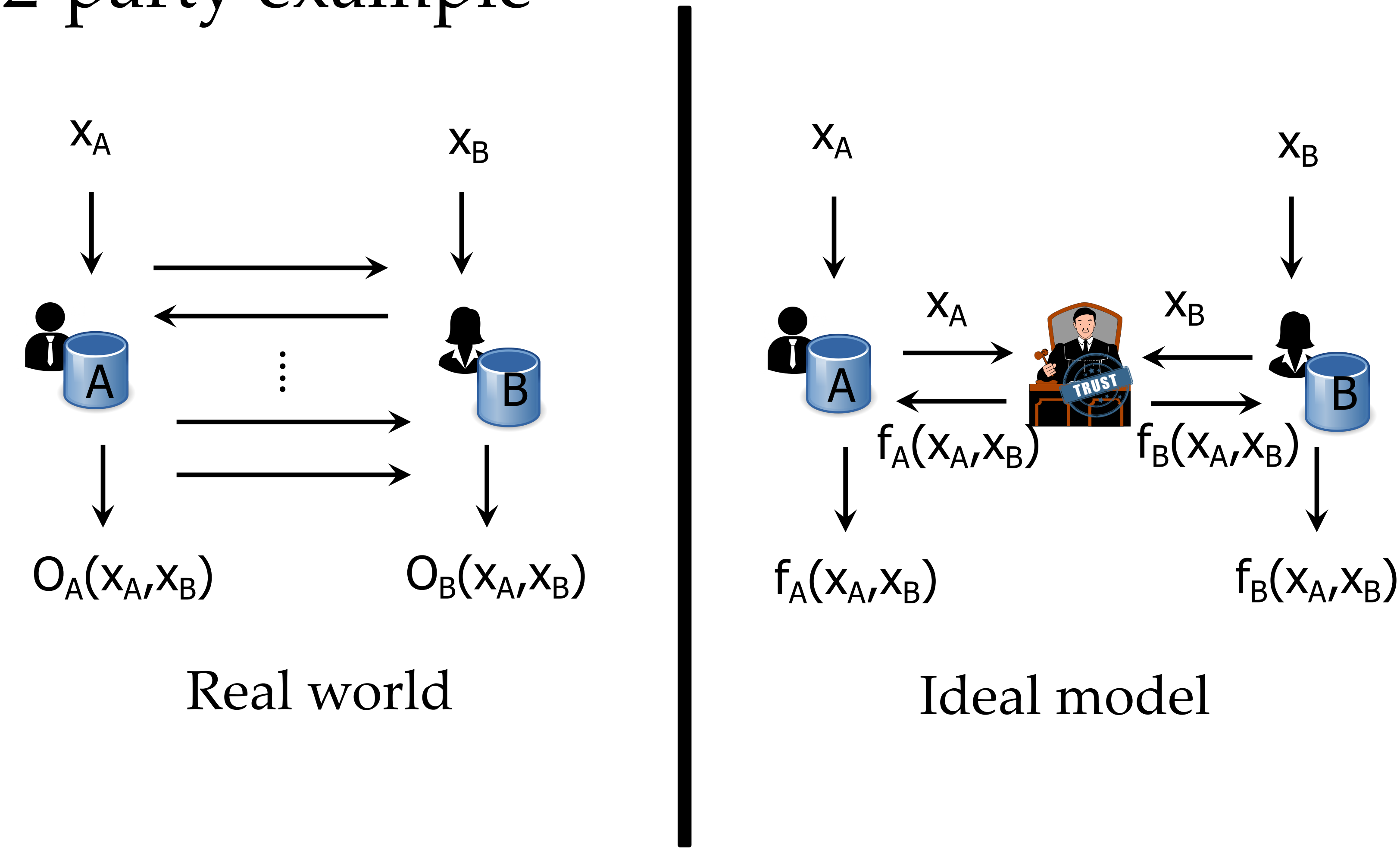
Multi-party Secure Computation

- Motivated use cases:
  - Can we have an auction without auctioneer?
  - Hospitals which cannot share their patient records with anyone want to mine on the combined data.

- Emulate a source of **trusted** computation
  - It will not "leak" a party's information to others
  - And it will not cheat in the computation

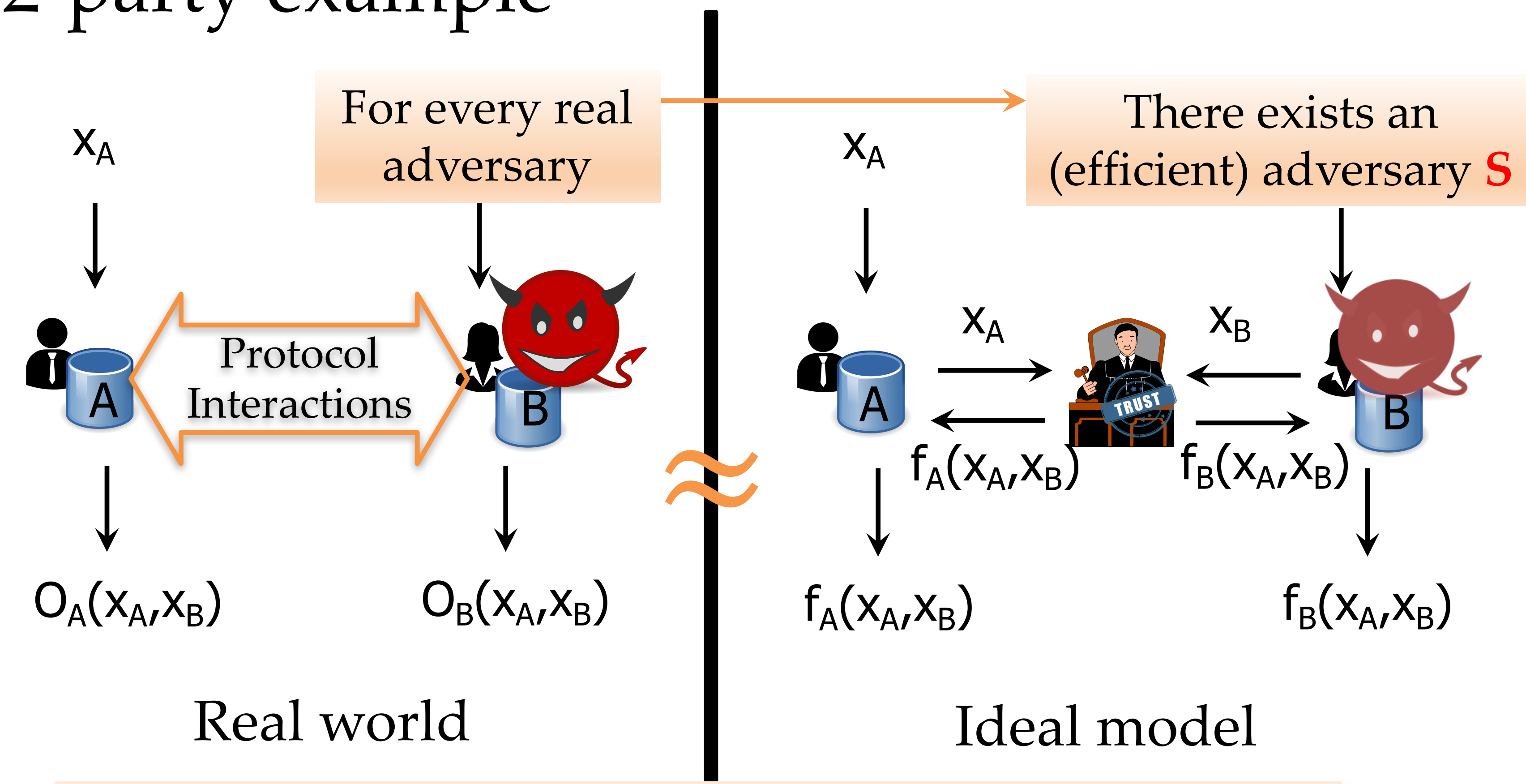
Simulation-based MPC

- 2-party example



Simulation-based MPC

- 2-party example



$\{\text{view}_B(\text{real protocol}), O_A, O_B\} \approx \{S_B(x_B,f_B), f_A, f_B\}$

Simulation-based MPC

- Protocol for computing $f(X_A,X_B)$ betw. A and B is **secure** if there exist efficient simulator algorithms S_A and S_B such that for all input pairs (x_A,x_B)

  $\{view_A(\text{real protocol}), O_{A,} O_B\} \approx \{S_A(x_A,f_A), f_A, f_B\}$

  $\{view_B(\text{real protocol}), O_{A,} O_B\} \approx \{S_B(x_B,f_B), f_A, f_B\}$

- Correctness: $(O_A,O_B) \approx f(x_A,x_B)$
  - In the ideal model, the function is always computed correctly
  - Thus, the same is true in the real-model

Adversary Model

- Computation power:
  - Prob. polynomial time v.s. all-powerful
- Adversarial behavior:
  - Semi-honest: follows the protocol, but tries to learn more (aka passive; honest-but curious)
  - Malicious: deviates from the protocol in arbitrary ways
- Corruption behavior:
  - Static: set of corrupted parties fixed at onset
  - Adaptive: can choose to corrupt parties at time during computation
- Number of corruptions:
  - Honest majority v.s. unlimited corruptions

Feasibility

- Any multiparty functionality can be securely computed
  - For any number of corrupted parties: security with abort is achieved, assuming enhanced trapdoor permutations [Yao,GMW]
  - With an honest majority: full security is achieved, assume private channels only [BGW,CCD]

Public-key Encryption

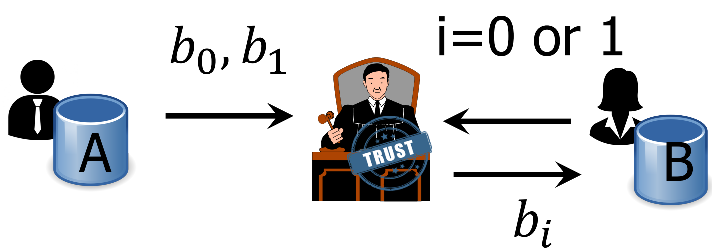
- Let (G, E, D) be a public-key encryption scheme
  - G : a key-generation algorithm $(pk, sk) \leftarrow G$
    - pk : public key; sk : secret key
    - Terms: m denotes plaintext; c denotes ciphertext
  - Encryption: $c = E_{pk}(m)$
  - Decryption: $m = D_{sk}(c)$
  - Concept of one-way function: knowing $c, pk, E_{pk}()$, it is still computationally intractable to find m

Construction Paradigms

- Passively-secure computation for two-parties
  - Use oblivious transfer to securely select a value
- Passively-secure computation with shares
  - Use secret sharing scheme such that data can be reconstructed from some shares
- From passively-secure protocols to actively secure protocols
  - Use zero-knowledge proofs to force parties to behave in a way consistent with the passively secure protocol

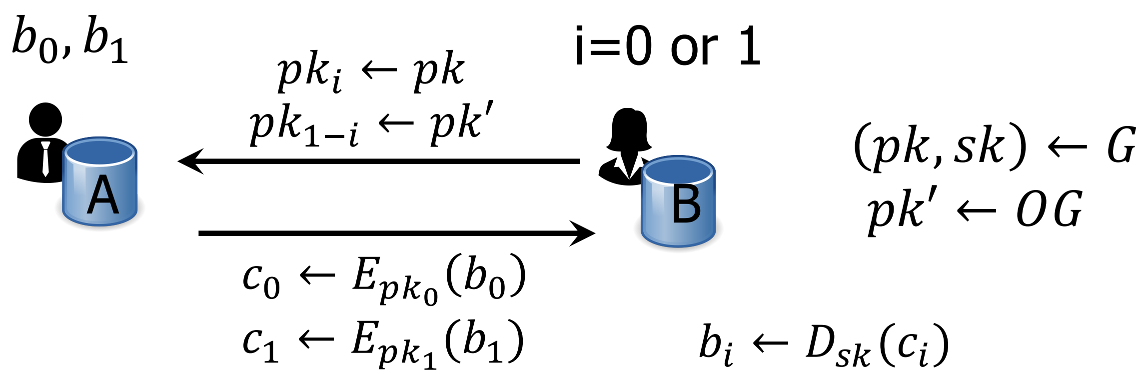
1-out-of-2 Oblivious Transfer (OT)

- A inputs two bits
- B inputs the index of one of A's bits
- B learns his chosen bit, A learns nothing
  - A does not learn <u>which</u> bit B has chosen; B does not learn the value of the bit that he did <u>not</u> choose



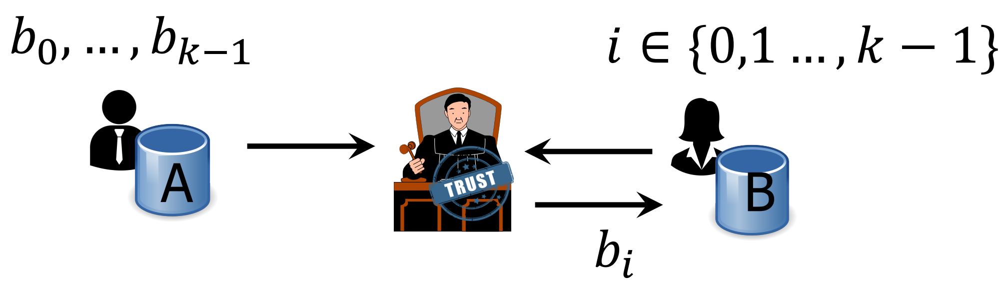
Semi-Honest OT

- Let (G, E, D) be a public-key encryption scheme
  - G : a key-generation algorithm $(pk, sk) \leftarrow G$
    - Assume that a pk can be sampled without knowledge of its sk [oblivious key generation, e,g,.El-Gamal encryption]: $pk \leftarrow OG$
  - Encryption: $c = E_{pk}(m)$
  - Decryption: $m = D_{sk}(c)$



Generalization [min-assignment 2]

- Can define 1-out-of-k oblivious transfer
- How?



General GMW Construction

[Goldreich-Micali-Wigderson]

- For simplicity – consider 2-party
  - Let f be the function that the two parties wish to compute over their inputs (a,b): f(a,b)

- Idea:
  - Represent f as an arithmetic circuit with addition and multiplication gates
  - Aim to compute gate-by-gate, revealing only random shares each time

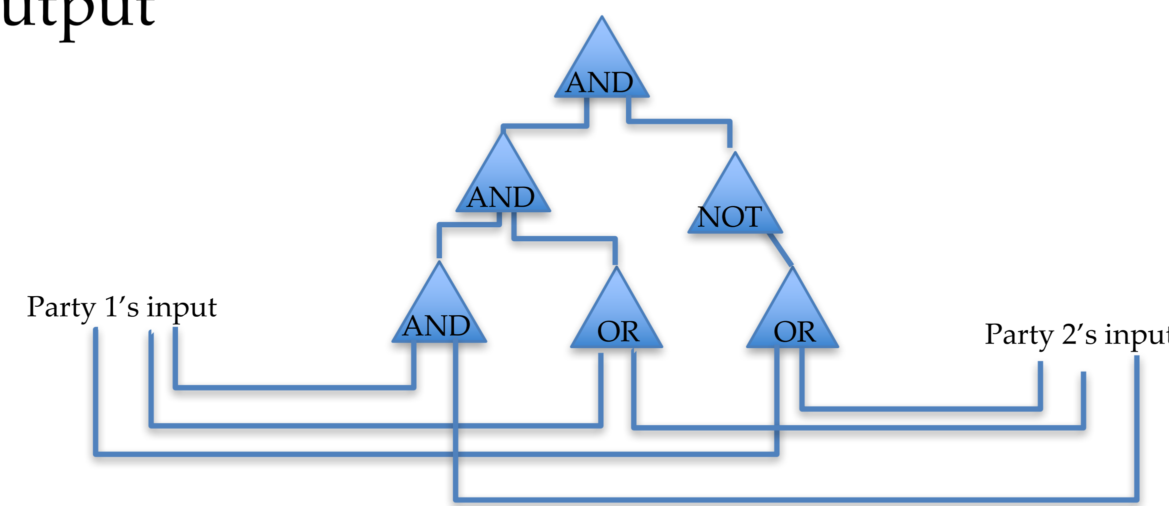
Random Shares Paradigm

- Let a be some value:
  - Party 1 holds a random value a_1
  - Party 2 holds $a + a_1$
  - Note that without knowing a_1, $a + a_1$ is just a random value revealing nothing of a
  - We say that parties hold random shares of a

- The computation will be such that all the intermediate values are random shares (and so they reveal nothing)

Circuit Computation

- Stage 1: each party randomly shares its input with the other party
- Stage 2: compute gates of circuits as follows
  - Given random shares to the input wires, compute random shares of the output wires
- Stage 3: combine shares of the output wires in order to obtain actual output



Addition Gates

- Input wires to gate have values a and b:
  - Party 1 has shares a1 and b1
  - Party 2 has shares a2 and b2
  - Note: a1+a2=a, b1+b2=b
- To compute random shares of output c=a+b
  - Party 1 locally computes c1=a1+b1
  - Party 2 locally computes c2=a2+b2
  - Note: c1+c2 = a1+b1+a2+b2 = a+b=c

Multiplication Gates

- Input wires to gate have values a and b:
  - Party 1 has shares a1 and b1
  - Party 2 has shares a2 and b2
  - Wish to compute c = ab = (a1+a2)(b1+b2)

- Party 1 knows (a1,b1)
- Party 2's values (a2,b2) are unknown to Party 1, but there are only 4 possibilities
  - (depending on correspondence to 00,01,10,11)

Multiplication Gates (cont.)

- Let r be a random bit chosen by Party 1
- Party 1 prepares a table as follows:
  - Row 1: ab+r when a2=0, b2=0
  - Row 2: ab+r when a2=0, b2=1
  - Row 3: ab+r when a2=1, b2=0
  - Row 4: ab+r when a2=1, b2=1

- For example
  - Assume a1=0,b1=1
  - Assume r=1

Row	Party 2's shares	Output Value
1	a2=0, b2=0	(0+0)(1+0)+1=1
2	a2=0, b2=1	(0+0)(1+1)+1=1
3	a2=1, b2=0	(0+1)(1+0)+1=0
4	a2=1, b2=1	(0+1)(1+1)+1=1

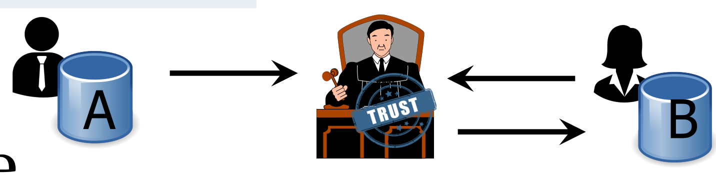
The Gate Protocol

- The parties run a 1-out-of-4 OT

Row	Party 2's shares
1	a2=0, b2=0
2	a2=0, b2=1
3	**a2=1, b2=0**
4	a2=1, b2=1

i =1,or 2, or **3**, or 4

Row	ab+r value
1	(0+0)(1+0)+1=1
2	(0+0)(1+1)+1=1
3	(0+1)(1+0)+1=0
4	(0+1)(1+1)+1=1

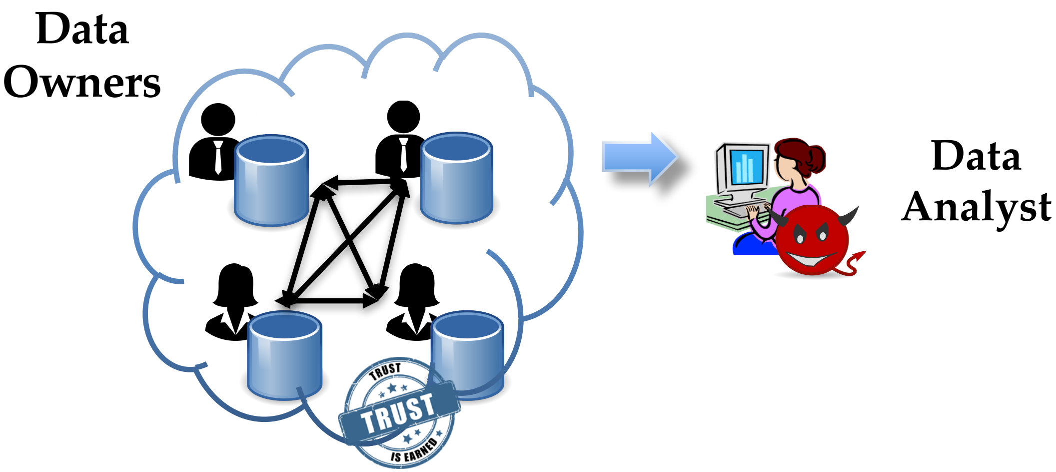


row_3 = ab+r

- For example
  - Assume a1=0,b1=1
  - Assume r=1

Challenges in Practice

- (m+1)-party MPC protocols are usually expensive



- Data owners need to be online

Challenges in Practice

- Security/privacy proofs can be tricky
  - Even for stand-alone crypto/DP mechanisms [BR06, LSL17]
  - Hybrid approach is vulnerable to faulty proofs [HMFS17]

//NoisyMax [DR14]**:**

For $i = 1, \ldots, L$: $\boxed{[c'_i] \leftarrow [c_i] + [\eta_i]}$, where $\eta_i \sim Lap(\frac{1}{\epsilon})$

Release $argmax_{i\in[1,L]}\ c'_i$

//Release counts for $i \leq 10$:

~~For $i = 1, \ldots, 10$: $[c'_i] \leftarrow [c_i] + [\eta_i]$, where $\eta_i \sim Lap$~~$(\frac{\Delta}{\epsilon})$

Release $(c'_1, c'_2, \ldots, c'_{10})$

2ϵ-DP

[xxx] denotes cipher text

Challenges in Practice

- Performance optimization is non-trivial

//Accurate Histogram Publication (AHP) [ZCXMX14]:

For $i \in [1, L]$: $[c_i'] \leftarrow [c_i] + [\eta_i]$, where $\eta_i \sim Lap(\frac{1}{\epsilon_1})$

$([c_{i_1}'], \ldots, [c_{i_L}']) \leftarrow Sort(Threshold([c_1'], \ldots, [c_L']))$

$C \leftarrow Cluster([c_{i_1}'], \ldots, [c_{i_L}'])$

For $C_i \in C$: $[C_i] \leftarrow \Sigma_{c_j \in C_i}[c_i']/|C_i|$

For $i \in [1, L]$:$[c_L''] \leftarrow [C_i] + [\eta_i]/|C_i|$, where $\eta_i \sim Lap(\frac{1}{\epsilon_2})$

Release $(c_i'', \ldots, c_L'')$

EMP toolkit
800s for a dataset of size 33k
→ 3.4x less time with optimization!!

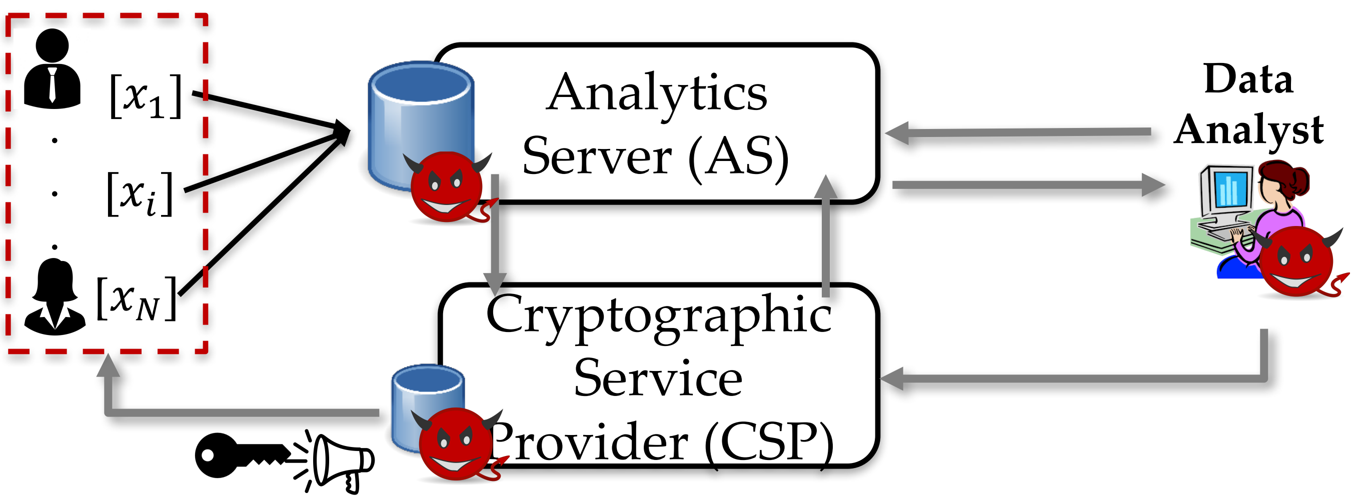
Outline

- Part I: Local DP
  - Randomized Response
  - RAPPOR
  - Limitations of Local DP
- Part II: Marrying DP with Crypto
  - Secure Multi-party computation
  - Crypte
- Upcoming Papers and Annoucements

Crypte

- ~~(m+1)-party MPC protocols are usually expensive~~
- Data owners need to be online
- Secu
- Perf

2-Server Model with minimal trust assumption
- Computationally bounded adversary
- Semi-honest behavior and non-collusion

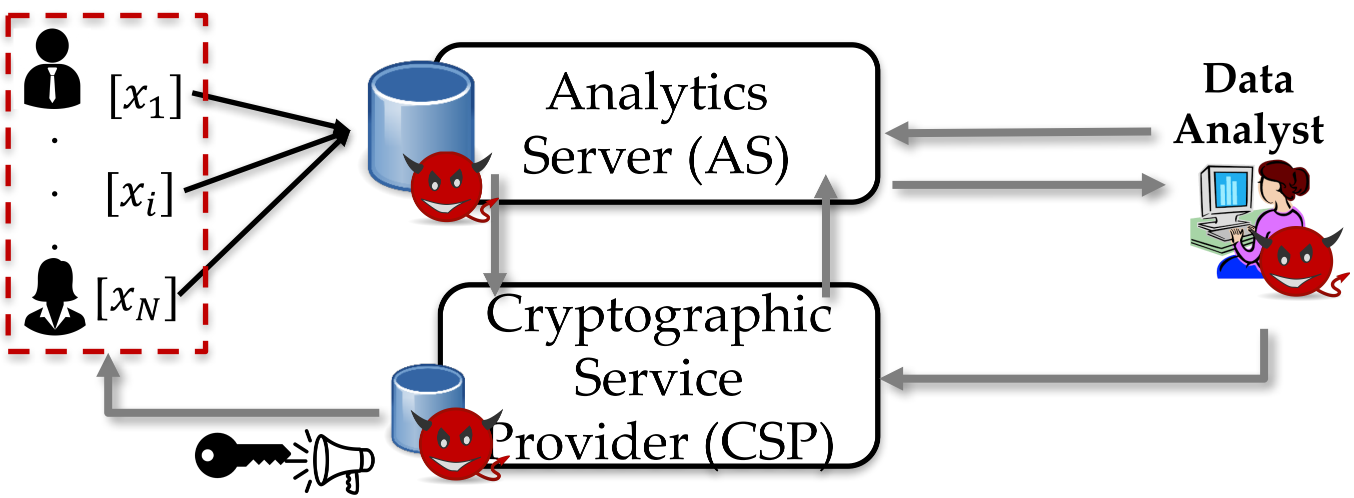


Crypte

- (m+1)-party MPC protocols are usually expensive
- ~~Data owners need to be online~~
- Securi[…]
- Perfo[…]

Data owners are offline

- CSP acts on behalf of data owners
- CSP is minimally involved in the protocols



Homomorphic Encryption

- Allows computations on the ciphertext without knowing the secret key, meanwhile ensures that the decryption of the resulting ciphertext is the same as the computations over the plaintext.

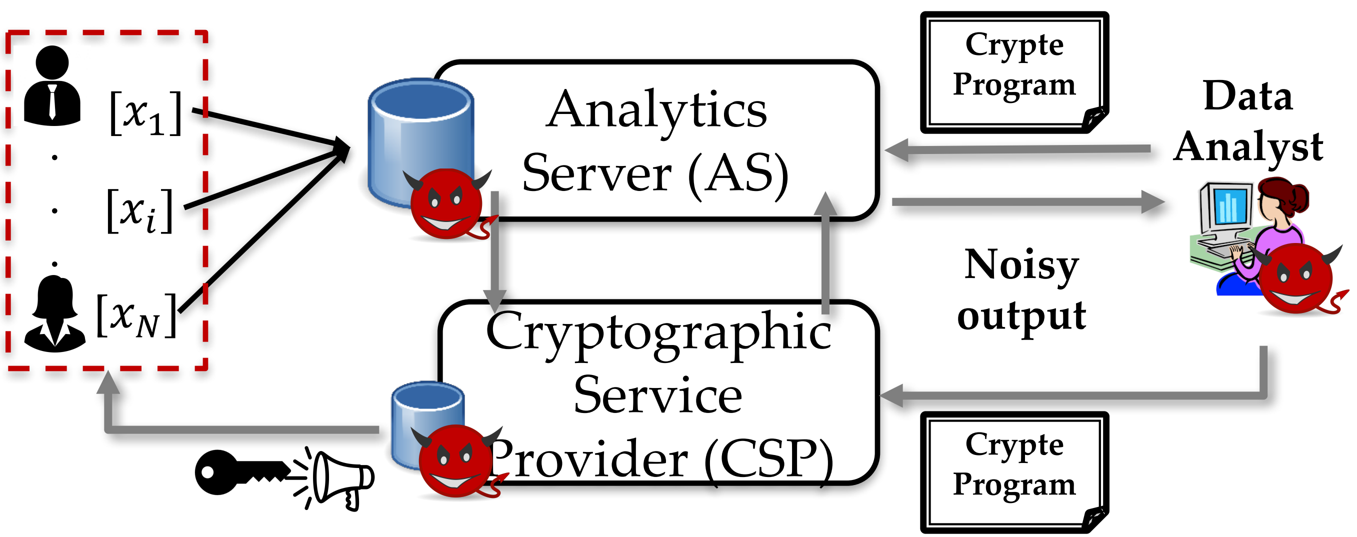
- Development
  - Idea about privacy homomorphism was proposed [RAD78]
  - Partially Homomorphic Encryption Schemes [RSA78] [Paillier99]
  - 1st generation FHE based on ideal lattice (bootstrapping) [Gen09b]
  - 2nd generation FHE based on RLWE (key/modulus switch) [BV11b][BGV12]
  - 3rd generation FHE based on LWE (approximate eigenvector) [GSW13]

Crypte

- (m+1
- Data
- ~~Security/privacy proofs can be tricky~~
- Performance optimization is non-trivial

Programming Framework (logical)

- Compile high level operators down to black box
- Prove security/privacy in modular fashion

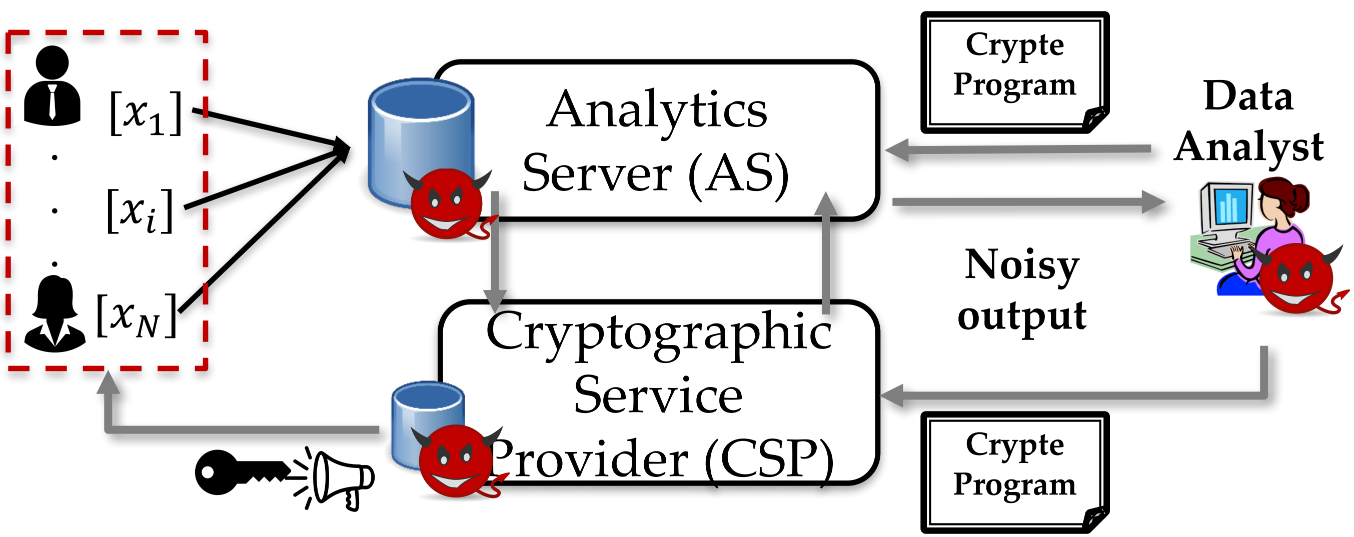


Crypte

- (m+1)-party MPC protocols are usually expensive
- Data c
- Securi
- ~~Performance optimization is non-trivial~~

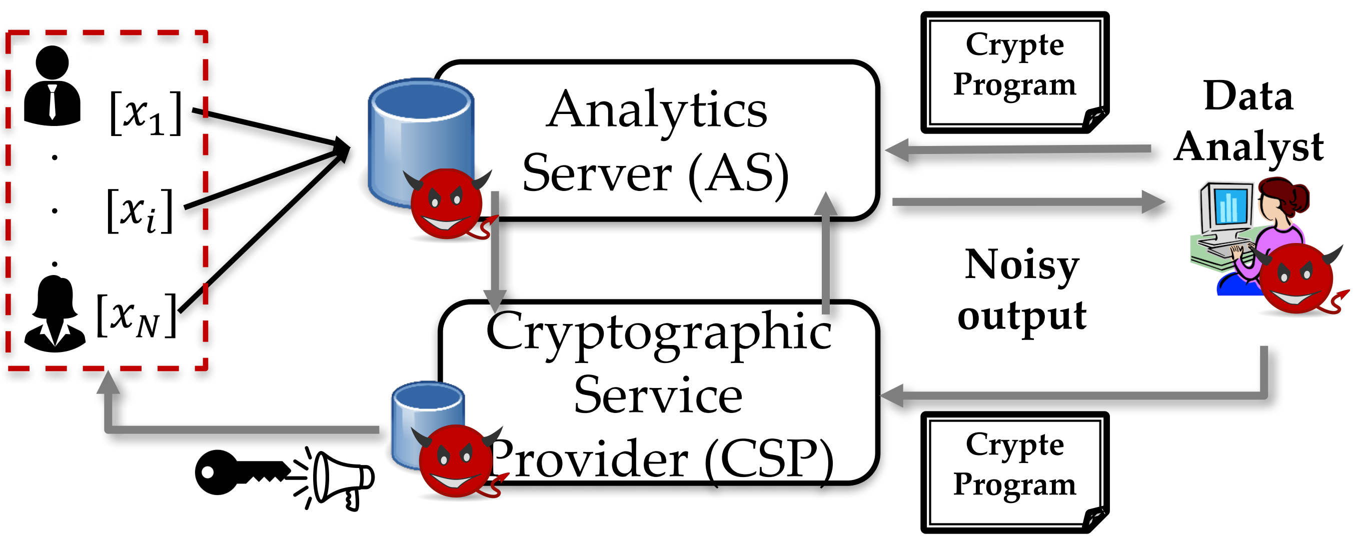
Built-in Performance Optimization

- DP-index optimization
- Crypto-engineering optimization

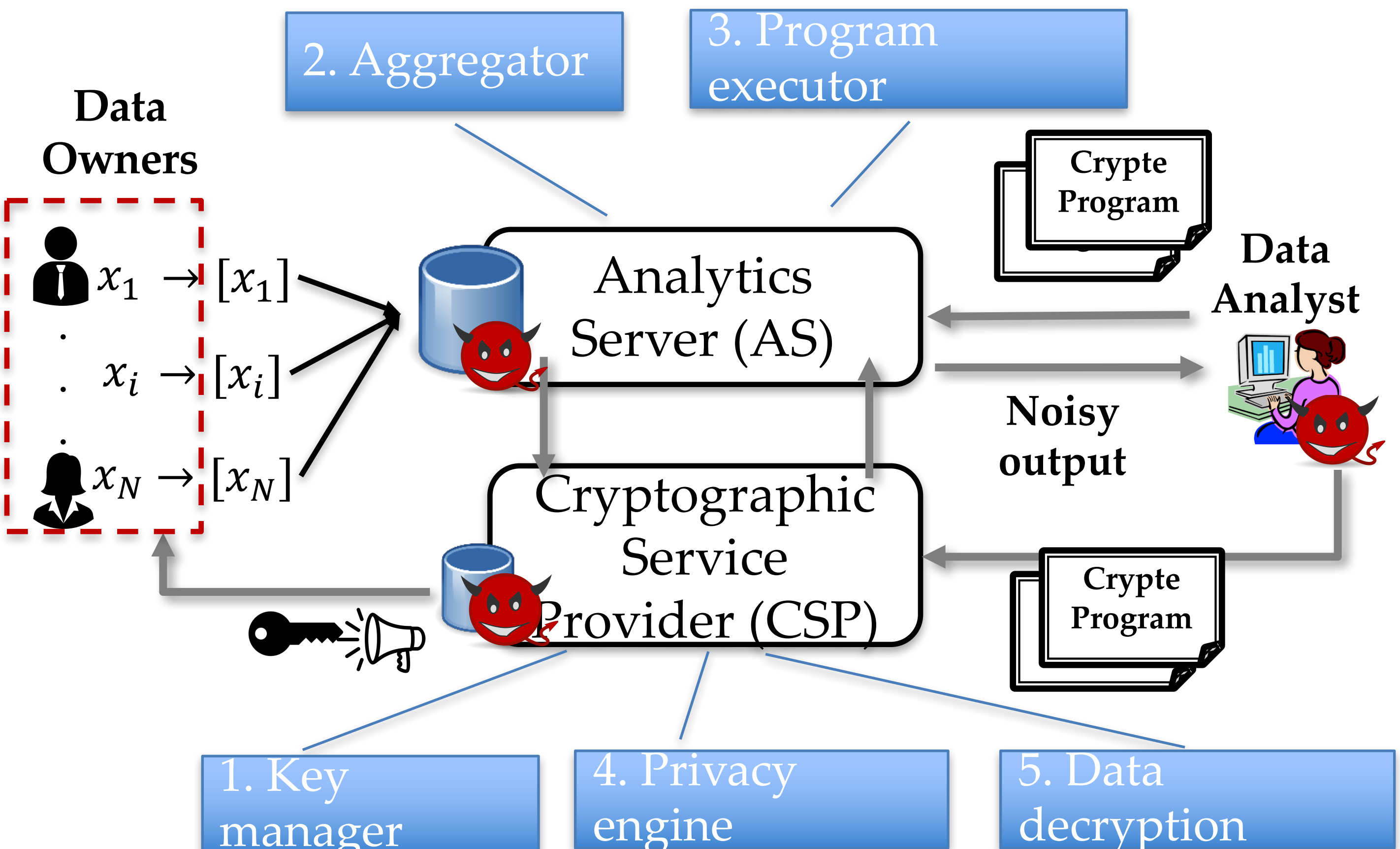


Design Principles in Crypte

- 2-server model with minimal trust assumptions
- Data owners are offline
- Programming framework
- Built-in performance optimization

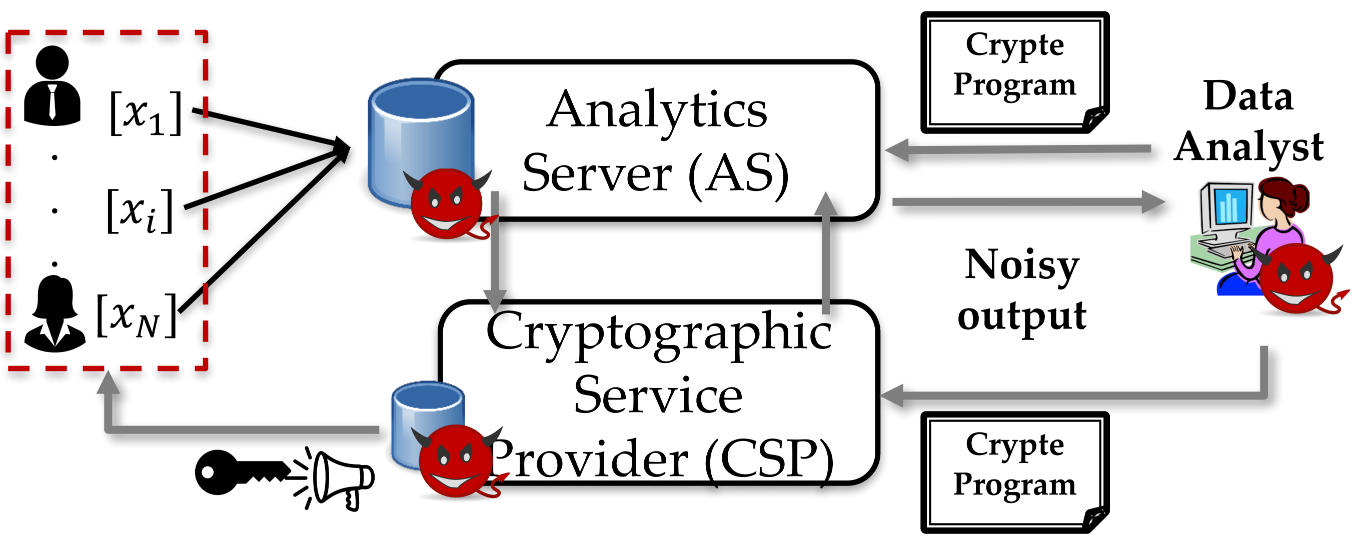


Crypte Overview



Design Principles

- 2-server model with minimal trust assumptions
- Data owners are offline
- Programming framework
- Built-in performance optimization



Programming Framework

- Logical program: a sequence of operators

Crypte Program

Transformation operators: $[D] \rightarrow [V]/[c]$

E.g. $\sigma_{condition}()$, $\pi_A()$, $count()$, $\gamma_A^{count}()$

Measurement operators: $[V] \rightarrow \tilde{V}$

E.g. $Lap()$, $NoisyMax()$

Postprocessing operators: $\tilde{V} \rightarrow \tilde{O}$

E.g. ≥ 0, $Post_{c.d.f.}$

- Views of AS and CSP
  - AS sees either encrypted values or noised values in the clear
  - CSP always sees noised values (encrypted/in the clear)

Program Example

- Database schema: *Age(A), Gender(G), NativeCountry(N), Race(R)*
- P1: Compute the cumulative distribution of *Age* ranged [1,100]

For $i \in [1,100]$:

$$\hat{c}_i \leftarrow Lap_{\epsilon_i,\Delta=1}\left(count\left(\sigma_{Age\in(0,i]}\left(\pi_{Age}\left(\tilde{D}\right)\right)\right)\right);$$

$$output \leftarrow post_{c.d.f.}([\hat{c}_1, \dots, \hat{c}_{100}])$$

DP noised

Encrypted

Program Example

- Database schema: *Age(A), Gender(G), NativeCountry(N), Race(R)*
- P5: Count the no. of male employees of Mexico having age 30

$$\hat{c} \leftarrow Lap_{\epsilon,\Delta=1}\left(count\left(\sigma_{A=30 \wedge G=M \wedge N=Mexico}\left(\pi_{A,G,N}(\widetilde{D})\right)\right)\right);$$

- P7: Count the no. of age values having at least 100 records

$$\hat{c} \leftarrow Lap_{\epsilon,\Delta=2}\left(count\left(\sigma_{Count\in[100,m]}\left(\gamma_A^{count}\left(\pi_A(\widetilde{D})\right)\right)\right)\right);$$

DP noised

Encrypted

Implementation Details

$$\hat{c}_{30} \leftarrow Lap_{\epsilon_{30},\Delta=1}\left(count\left(\sigma_{Age\in(0,30]}\left(\pi_{Age}\left(\widetilde{D}\right)\right)\right)\right);$$

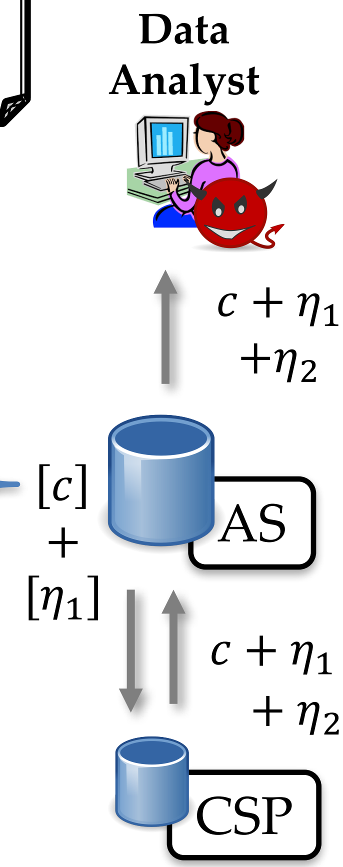
A	G	N	R	Bit
[18]				[1]
[35]				[1]
…				…
[42]				[1]
[23]				[1]

A	Bit
[18]	[1]
[35]	[1]
…	…
[42]	[1]
[23]	[1]

A	Bit
[18]	[0]
[35]	[1]
…	…
[42]	[1]
[23]	[0]

[0]	[0]	…	[1]	…	[0]	[0]	[0]

18th position



Linear Homomorphic Encryption: Dec[[x]+[y]]=x+y

Alternative Implementations

- Secret share based MPC protocol
  - 2 servers do the similar amount of work
- Joint noise generation [DKMMN06, NH12]
  - More expensive MPC protocol
- Other improvements:
  - Data representation: multi-attribute one-hot-encoding
  - Optimized HE scheme or GC

All are possible, due to the separation of logical programming framework and underlying physical implementation!

Security Sketch (Semi-honest)

For $i \in [1,100]$:

$$\hat{c}_i \leftarrow Lap_{\epsilon_i,\Delta=1}\left(count\left(\sigma_{Age\in(0,i]}\left(\pi_{Age}\left(\widetilde{D}\right)\right)\right)\right);$$

$output \leftarrow post_{c.d.f.}([\hat{c}_1, \dots, \hat{c}_{100}])$

Program P

Protocol Π

Satisfy SIM-CDP [MVRV09]

Exists PPT Sim_{AS}, Sim_{CSP}, such that

$$Sim_{AS}(P^{CDP}(D,\epsilon)) =_c \begin{pmatrix} View_{AS}^{\Pi}(P,D,\epsilon), \\ Output^{\Pi}(P,D,\epsilon) \end{pmatrix}$$

AS

$$Sim_{CSP}(P^{CDP}(D,\epsilon)) =_c \begin{pmatrix} View_{CSP}^{\Pi}(P,D,\epsilon), \\ Output^{\Pi}(P,D,\epsilon) \end{pmatrix}$$

CSP

r.v.: the output of running P in CDP model

Security Sketch (Semi-honest)

For $i \in [1,100]$:

$$\hat{c}_i \leftarrow Lap_{\epsilon_i, \Delta=1}\left(count\left(\sigma_{Age \in (0,i]}\left(\pi_{Age}\left(\widetilde{D}\right)\right)\right)\right);$$

$output \leftarrow post_{c.d.f.}([\hat{c}_1, \dots, \hat{c}_{100}])$

Program P

Protocol Π

Satisfy SIM-CDP [MVRV09]

Encryption

…

Garbled circuit

Laplace operator

AS

CSP

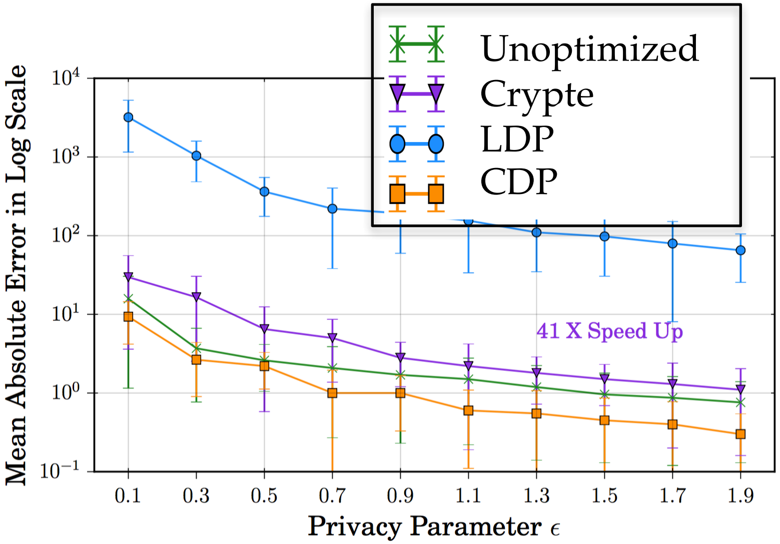
Composition theorem

[G. Oded. Foundations of Cryptography]

Design Principles

- 2-server model with minimal trust assumptions
- Data owners are offline
- Programming framework
- Built-in performance optimization
  - DP-index Opt
    - E.g. index on *NativeCountry*
  - Crypto-engineering Opt
    - DP range tree
    - Pre-computation
    - Offline processing

P5: 1201.12s → 29.21s



Evaluation

- Dataset of 32,651 rows and 7 Crypte programs
- Accuracy:
  - **The same order** as that of CDP implementation
  - 2-orders of smaller error than that of LDP implementation
- Performance:
  - Optimizations improve the performance by up to **5667×**
  - A large class of programs execute within **5 mins** and scale linearly with the data size
  - AS performs majority of the work for most programs

Take-away and Future work

Key: Separation of logical programming framework and underlying physical implementation!

- User-specified query in a high-level language
- A larger class of programs
- Malicious setting

Key: Leaky crypto for performance-privacy trade-off!

- DP for Crypto to improve performance
  - Computation [B**H**EMR17]
  - Access pattern [TDG16, WCM18]
  - Communication [HLZZ15, TGLZZ17, LGZ18]

Summary

- Part I: Local DP
  - Randomized Response
  - RAPPOR
  - Limitations of Local DP
- Part II: Marrying DP with Crypto
  - Secure Multi-party Computation
  - Crypte

Discussion Time

Paper Readings

- Week 7
  - 2a. Prochlo
  - 2b. Orchard
  - 2c. EncryptedDB

- Week 8
  - 2d. Collecting data jointly under LDP
  - 2e. DJoin
  - 2f. Shrinkwrap

Announcement

- Assignment 1
  - Release after Wed session
  - Latex file will be available
  - Submit pdf on Learn, by Feb 22, 11pm

- Feedback on projects will be emailed before/during the reading week

Slides Credits

- http://sigmod2017.org/wp-content/uploads/2017/03/04-Differential-Privacy-in-the-wild-2.pdf
- https://www.cs.utexas.edu/~shmat/courses/cs380s_fall09/15smc.ppt
- http://www.mathcs.emory.edu/~lxiong/cs573_s12/share/slides/12crypt.pdf